

How to upgrade your cubes from 9.x to 10 and turn on optimized parallel in-memory analytics

Karthik K Iyengar

Agenda

Introduction

What is MicroStrategy PRIME?

Key Features of PRIME

MTDI Interface

Upgrading your 9.x Cubes to PRIME

Governing Rules, VLDB Properties and Registry Settings

Performance Tips and Tricks

Demo

Summary and Takeaways, Q&A

What is MicroStrategy PRIME?

MicroStrategy PRIME is the next generation of MicroStrategy's in-memory technology

- ❑ Embeds an **in-memory, column-oriented, distributed, analytic** data store...
- ❑ ...with a **tightly coupled, state of the art visualization and interaction engine**, to enable speed of thought performance for thousands of users across very large and complex datasets

Parallel



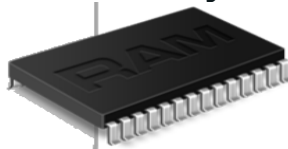
Data is partitioned and processed in parallel

Relational



Load data for an entire application, with Entity-Relation Model Semantics, into memory

In-memory



Data stored in-memory for fast query response time

Analytics Engine

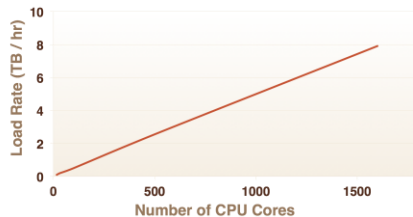


Focus on read-mostly analytics; not an ACID Compliant general purpose DB

PRIME - Parallel Relational In-Memory Engine

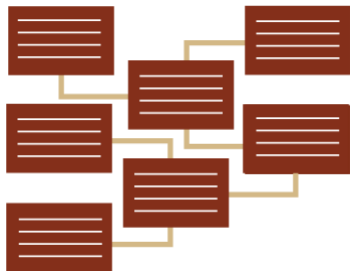
PARALLEL

Linear scalability to 1,000s of CPUs



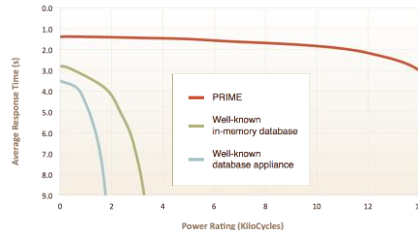
RELATIONAL

Flexible schema & Partitioned data



IN-MEMORY

3x to 10x faster
7x to 20x more users



ENGINE

Tightly-coupled interactive exploration



Key Features

Three Key Dimensions that PRIME evolves over 9.x Cubes

Larger Data Volumes:

- ❑ Current OLAP in-memory technology limited to 2 billion rows.
- ❑ MicroStrategy PRIME datasets can be divided into partitions. **Each partition can have up to 2 billion rows.**

Broader Schema Support:

- ❑ Current in-memory cubes are restricted to a single-star schema.
- ❑ In-memory cubes are closer to a cached report result set.
- ❑ MicroStrategy PRIME extends to **multiple fact tables**; fact tables with varying grains; M-M relationship tables; Entity-Relation Model Semantics.
- ❑ MicroStrategy PRIME is closer to the raw data staged in-memory.

Broader Analytical Functionality:

- ❑ Current in-memory cubes do not support full Filter and Metric functionality of the MicroStrategy platform.
- ❑ MicroStrategy PRIME extends to “**Multi-pass**” **analytics** with multiple levels of aggregation and filtering.

Data Distribution in PRIME

- ❑ One of the most important benefits of PRIME is the ability to **Partition** and **Process Data in Parallel**
- ❑ Partitioning allows **distribution of data across multiple cores on a single box** to enable parallel processing of information
- ❑ Each data partition is in a “**shared nothing**” architecture and works only with its corresponding CPU core.
- ❑ Type of Aggregations that can be performed on the raw data include –
 - ✓ Distributive Functions – *SUM, MIN, MAX, COUNT*
 - ✓ Semi-Distributive Functions – *STD DEV, VARIANCE*
 - ✓ Scalar Functions – *Add, Greatest, Date/Time Functions, String Manipulation Functions*
 - ✓ *DISTINCT COUNTs*
 - ✓ Derived Metrics using any of MicroStrategy’s in-built Functions

Partitioning Attribute and Number of Partitions

- ❑ PRIME currently supports only **one partitioning key/attribute for the entire dataset**. All tables that have the partition attribute will have their data distributed along the elements of that attribute.
- ❑ All Flavors of INT data types, STRING/TEXT data types as well DATE data types are supported as Partitioning Attributes.
- ❑ Guidelines for identifying a good partition attribute
 - ✓ Identify the largest FACT tables in the application since they influence the choice of the partition attribute
 - ✓ Ensure most number of partitions are accessed for any type of question that is asked of the application
 - ✓ Typically, **Number of Partitions = Number of Physical Cores available to the I-Server**. This rule maximizes CPU usage for best possible performance.
 - ✓ If a single column doesn't meet the criteria, consider adding a **PSEUDO column, like a ROWNUMBER** to the FACT table or to a Database View based on the FACT table

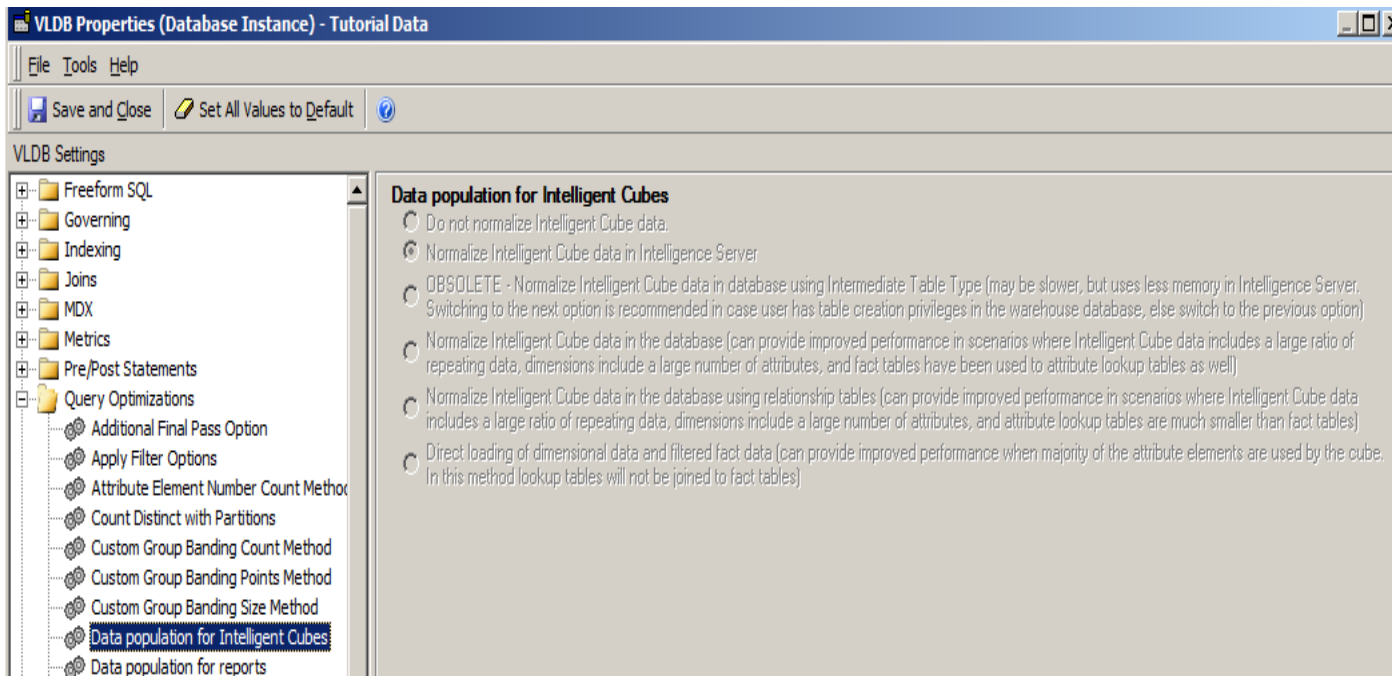
Data Storage in PRIME's In-Memory Engine

- ❑ Data is organized into 3 types of tables
 - ❑ **Lookup Tables** – Captures all the elements of the attribute from different tables in the PRIME dataset. Mostly used for analysis involving non-ID forms of the attribute, like filtering, calculations, etc.
 - ❑ **Relationship Tables** – Capture relationship between pairs of attributes as defined in the schema. Mostly used for queries involving joins, aggregation, filtering from lower level of detail to higher level. The tables are categorized as:
 - ❑ *ONE_TO_MANY* – For e.g. DAY:MONTH or QUARTER:YEAR
 - ❑ *MANY_TO_MANY* – For e.g. ITEM:CATALOG
 - ❑ **Fact Tables** – All Physical tables loaded into PRIME with Metric Mapping are stored internally as Metric Tables. These tables are used for KPI calculations.
 - ❑ *Index Pool* – Each metric table also has one associated index pool. These pools are used to store data structures that improve performance.

Upgrading your 9.x Cubes to
PRIME

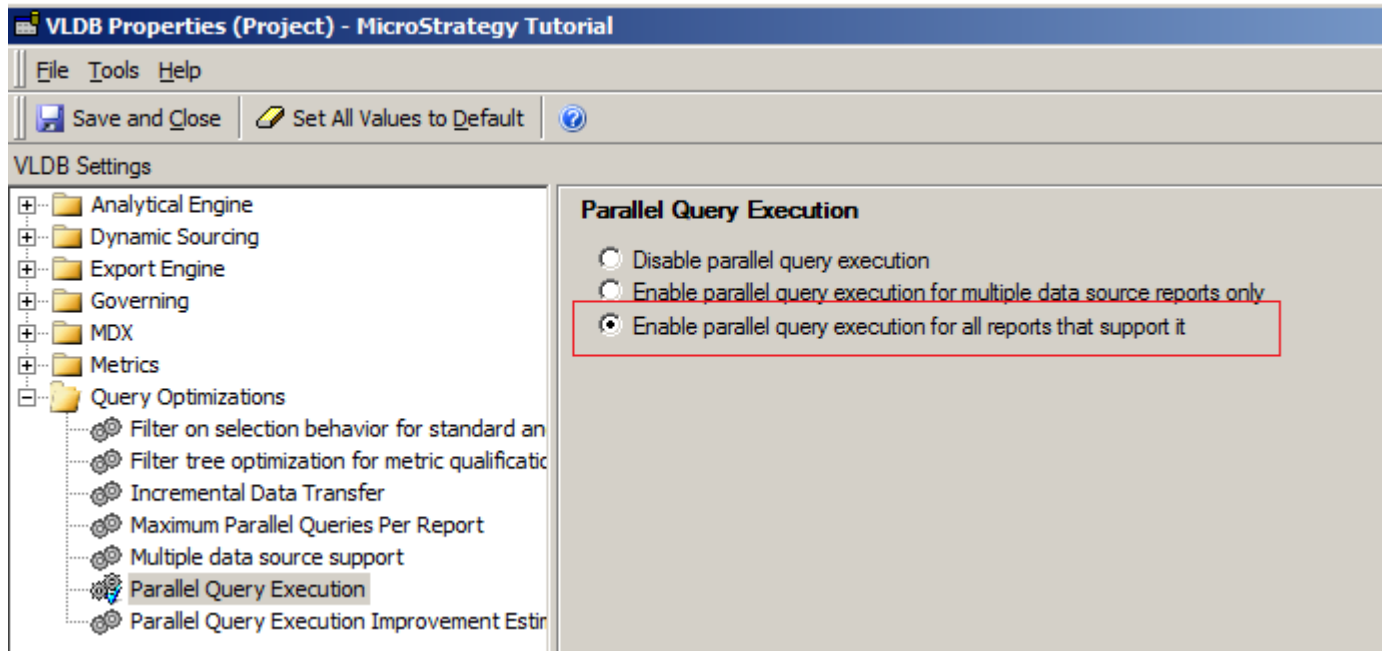
Configuring the 9.x Cubes for PRIME – Data Load from DWH

- ❑ In order to leverage the parallelism of v10 and apply it to the upgraded v9 I-Cubes, three steps need to be done:
 - ✓ **Step 1:** For pulling data from DWH, by default a single ODBC thread is used. In order to use parallel ODBC threads when pulling data from DWH, make sure the I-Cube VLDB Property is set to default “**Normalize Intelligent Cube data in Intelligence Server**” under “**Query Optimizations → Data population for Intelligent Cubes**”.



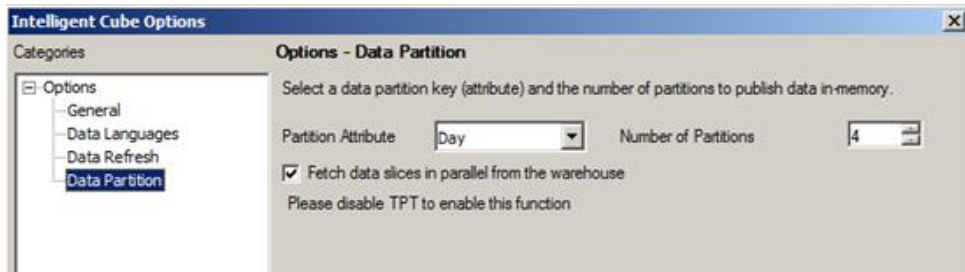
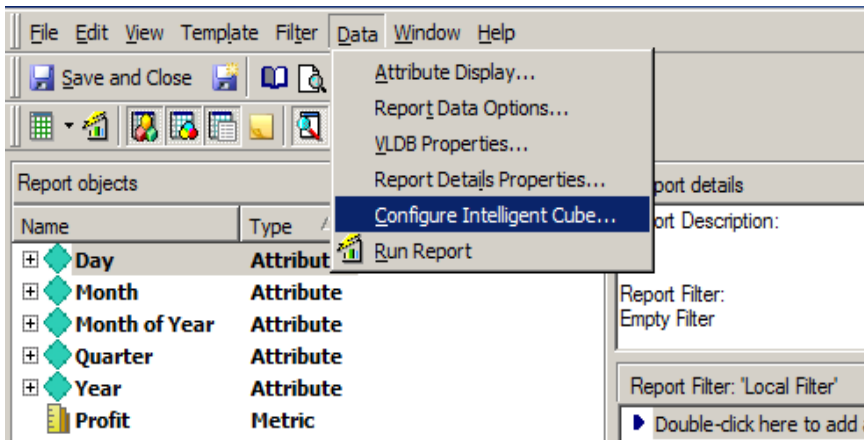
Configuring the 9.x Cubes for PRIME – Data Load from DWH

- ✓ **Step 2:** Enable parallel query execution for all reports that support it by checking the corresponding VLDB Property. This property is available under “*Project Configuration → Advanced → Project Level VLDB Settings → Query Optimizations*”



Configuring the 9.x Cubes for PRIME – Data Publish into Memory

- ✓ **Step 3:** For loading data into memory in parallel, the Intelligent Cube needs to be configured for parallelism.
 - ❑ Edit the I-Cubes and go to “**Data → Configure Intelligent Cube**”
 - ❑ Define the “**Partition Attribute**” and “**Number of Partitions**”. Check the “**Fetch data slices in parallel from the warehouse**” option.



MTDI Interface

Data Load into Memory – Multi Table Data Import

- ❑ Tables and Files can be imported directly into memory using the Multi Table Data Import web interface
- ❑ MTDI Datasets can be published/republished/incrementally refreshed similar to 9.x I-Cubes
 - ❑ Republishing allows user to choose which tables are to be refreshed and how.
 - ❑ Refresh Options include “**Add New Rows**”, “**Add New Rows and Update Existing Rows**”, “**Update Existing Rows**”
- ❑ MTDI Datasets can be scheduled
 - ❑ Both time-based and event-based schedules can be used for refreshes
- ❑ Security Filters
 - ❑ On Project Attributes – Same as in v9.x in-memory cubes
- ❑ “**Local Schema Managed Objects**” vs “**Project Schema Objects mapped into MTDI**”
 - ❑ When Objects defined in an MTDI Cube are not mapped to project schema objects, the MTDI objects remain “local” to the cube, and relationships need to be defined explicitly.
 - ❑ When Objects defined in the MTDI Cube are mapped to project schema objects, the objects remain “global”, and changes done to schema objects are reflected in MTDI as well.

MTDI Interface – Adding Tables

The screenshot displays the MTDI interface for adding tables. On the left, a dashed box with a plus sign and the text "Add a new table" is visible. In the center, three table cards are shown: "CUSTOMER_SLS", "CAL_DATA", and "INVENTORY_Q1_2014". The "INVENTORY_Q1_2014" card is highlighted with a blue border. Each card lists attributes and metrics. A red arrow points from the top right of the interface to the "All Objects View..." link. At the bottom right, there are buttons for "Update Dataset", "Save Progress", and "Cancel".

Preview

Search:

[All Objects View...](#) ? x

Table Selection:

- CUSTOMER_SLS**
 - Attributes: Customer Id
 - Metrics: Tot Dollar Sales, Tot Unit Sales, Tot Cost, Gross Dollar Sales
- CAL_DATA**
 - Attributes: County Id, County Desc
 - Metrics: County 2, Pop 2006, Pop 2007, Change
- INVENTORY_Q1_2014**
 - Attributes: Month Id, Item Id
 - Metrics: Boh Qty, Eon Qty

Bottom Bar:

INVENTORY_Q1_2014 Preview

Update Dataset Save Progress Cancel

MTDI Interface – Setting up the Partitions

All Objects View ? x

☐ Partition Attribute Automatic

Number of Partition 4

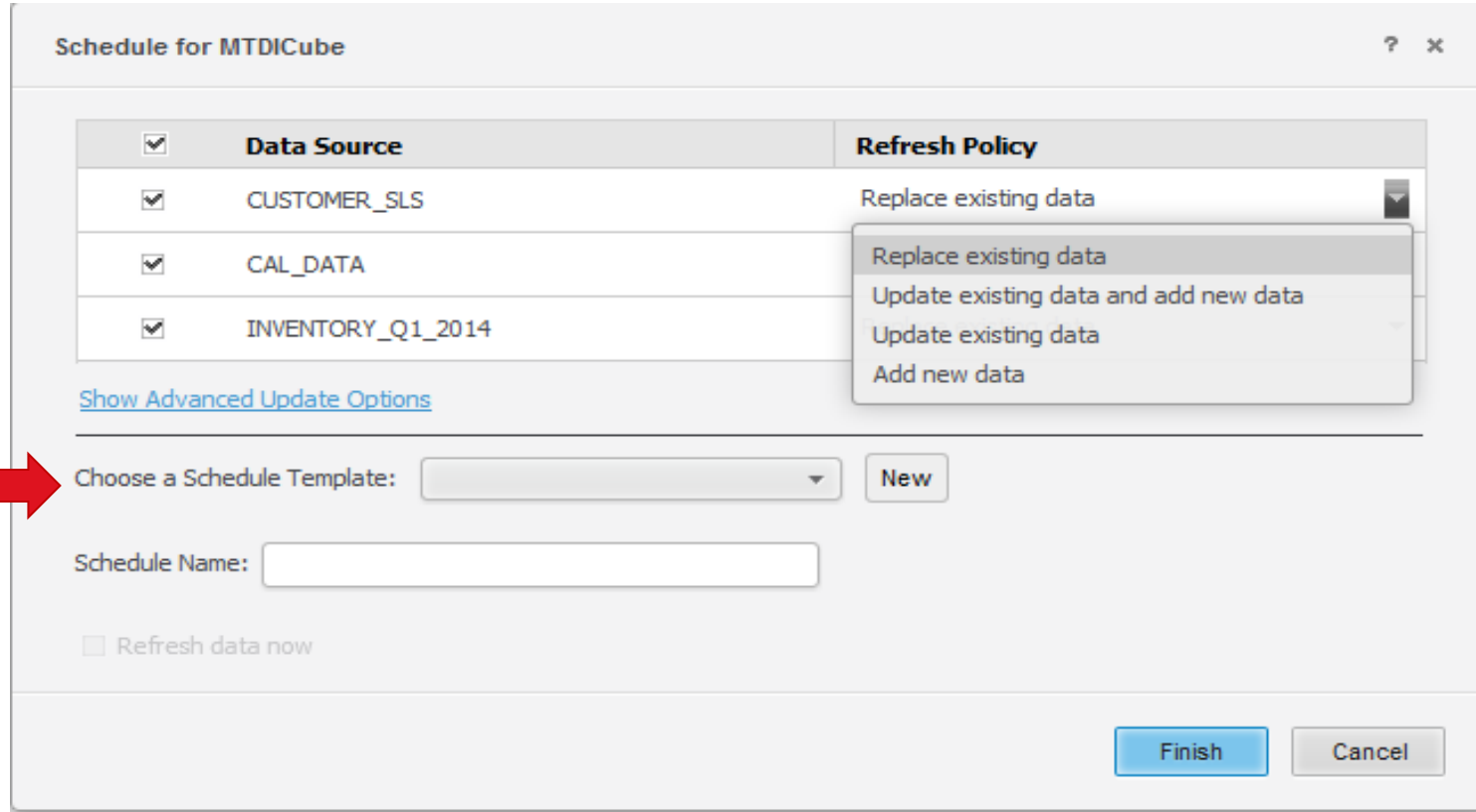
<input type="checkbox"/> Search Index	Attributes	Table
<input type="checkbox"/>	County Desc	1/3
	County Id	1/3
	Customer Id	1/3
	Item Id	1/3
	Month Id	1/3

Metrics	Table
Boh Qty	1/3
Change	1/3
County 2	1/3
Eoh Qty	1/3
Gross Dollar Sales	1/3
Pop 2006	1/3
Pop 2007	1/3
Tot Cost	1/3
Tot Dollar Sales	1/3
Tot Unit Sales	1/3

Save

Cancel

MTDI Interface – Cube Refresh Policy



Schedule for MTDICube ? x

<input checked="" type="checkbox"/> Data Source	Refresh Policy
<input checked="" type="checkbox"/> CUSTOMER_SLS	Replace existing data
<input checked="" type="checkbox"/> CAL_DATA	Replace existing data
<input checked="" type="checkbox"/> INVENTORY_Q1_2014	Update existing data and add new data

[Show Advanced Update Options](#)

Choose a Schedule Template:

Schedule Name:

☐ Refresh data now

Governing Rules, VLDB Properties and Registry Settings

Governing Settings and VLDB Properties

- Governing settings and VLDB Properties help tune the Analytical Engine for better performance

- ❖ VLDB Properties

- ❑ ***Project Configuration → Advanced → Project-Level VLDB Settings – Configure → Query Optimizations → Parallel Query Execution***
 - ✓ Option 3 – Enable parallel query execution for all reports that support it
 - ❑ ***Project Configuration → Advanced → Project-Level VLDB Settings – Configure → Query Optimizations → Maximum Parallel Queries per Report***
 - ✓ Increase this number from the default setting of 2

- ❖ Governing Settings

- ❑ ***Project Configuration → Intelligent Cubes → General → Maximum cube size allowed for download (MB):***
 - ✓ Increase this number from the default setting of 100
 - ❑ ***Project Configuration → Governing Rules → Import Data → “Maximum file size (MB)” and “Maximum quota per user (MB)”***
 - ✓ Increase this number from the default setting of 100

Registry Settings

- ❑ Always apply the latest patches, test builds, hot fixes released by MicroStrategy, and keep the configurations up-to-date.
- ❑ By making changes to the registry file (MSIREG.REG), the following settings can be used for performance improvements
 - ❑ `[HKEY_LOCAL_MACHINE\SOFTWARE\MicroStrategy\DSS Server\Castor]`
 - ❑ **"DisableAlternativeCSI"**=dword:00000000
 - ❑ This setting enables Alternative CSI. Changing the value to "dword:00000001" disables Alternative CSI
 - ❑ **"DeleteManyToMany"**=dword:00000001
 - ❑ This setting deletes many-to-many relationships if 1 table has 2 or more many-to-many relationship.
 - ❑ `[HKEY_LOCAL_MACHINE\SOFTWARE\MicroStrategy\DataImport\AutoDetect]`
 - ❑ **"NeedAutoDetect"**=dword: 00000000
- ❑ Any change to registry settings should be done carefully and a rollback plan should be in place.

Performance Tips and Tricks

CSI – Cube Subsetting Instruction

❑ What is CSI?

- ❑ The **Cube Subsetting Instruction** or CSI is generated whenever a report, document, or dashboard containing an Intelligent Cube or subset report is run. It indicates which attributes and metrics need to be brought back from the Intelligent Cube, along with any filter conditions that need to be applied.
- ❑ The way CSI is structured is very similar to SQL, with the exception that the data source is an Intelligent Cube instead of a database
- ❑ By closely reviewing the CSI, one can gain critical insights into how reports based out of Cubes will perform.

❑ CSI examples for OLAP and PRIME Cubes:

Example of CSI reading from an Intelligent Cube:

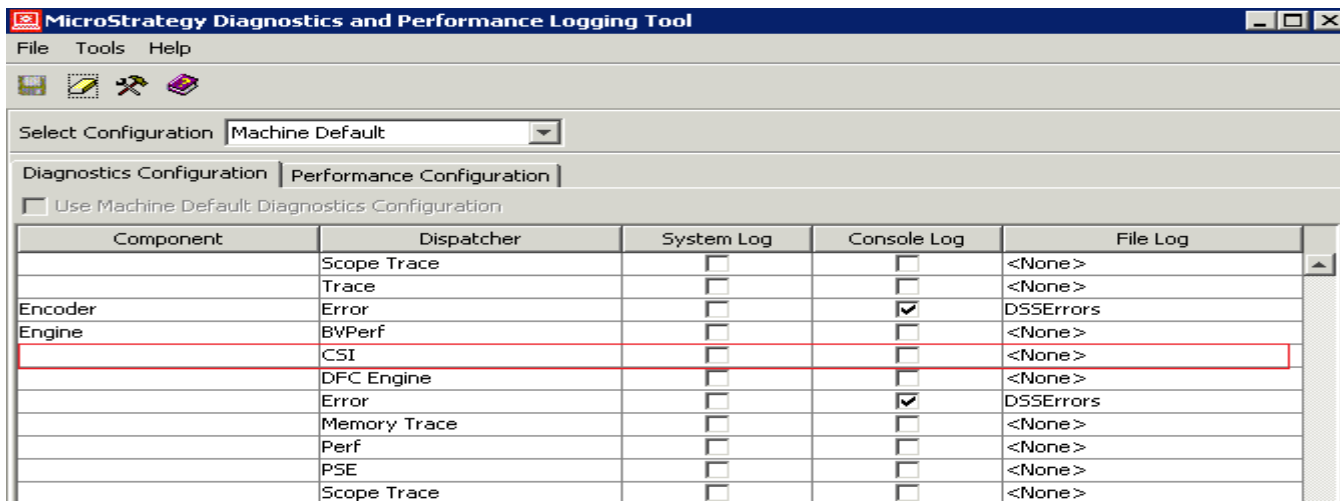
```
select [Customer Region]@[CUST_REGION_ID], [Customer Region]@[CUST_REGION_NAME],  
sum([Revenue])@{[Customer Region]}, sum([Profit])@{[Customer Region]} from Table 1  
where [Customer Region]@[CUST_REGION_ID] in (1, 2, 3)
```

Example of CSI reading from a PRIME Cube:

```
Tables Accessed: Table4:Call Center, LOOKUP_TABLE Table9:  
Subcategory,Quarter,Year,Region,Call Center,Category,M01, FACT_TABLE select [Call  
Center]@[CALL_CTR_ID], [Call Center]@[CENTER_NAME], [Year]@[YEAR_ID],  
sum([Table9.M01])@{[Call Center],[Year]} as [M01] from Cube02- Prime Cube by AE  
with Table Join Tree: Table9<[Year]@[YEAR_ID] in (2008.000000)>
```


How to activate the **Cube Subsetting Instruction (CSI)** Log in MicroStrategy 10

- ❑ Open the **MicroStrategy Diagnostics and Performance Logging** Tool. Navigate to the Engine Component with **CSI Dispatcher**.



- ❑ Create a new File Log and name it "CSI" (or any other name). Then click "Save".
- ❑ After saving the log destination, the File Log column should now read as "CSI" for the Engine Component in the CSI Dispatcher
- ❑ After running a report, document, or dashboard based on an OLAP or PRIME cube, the CSI log file can be found (depending on the selected installation path) at: "C:\Program Files (x86)\Common Files\MicroStrategy\Log"

MCE – Multi Cube Engine

❑ What is MCE?

- ❑ MCE trace helps provide backend analysis of **Multi-Cube Engine** activity. The Multi-Cube engine is the engine that handles dataset join on a report services document grid.
- ❑ **COOL TIP:** This trace can be used to obtain the SQL Query information from a Visual Insight Dashboard in 9.4.x as well.

- ❑ **Example:** For a document that contains three cubes and when the metrics comes from different cubes in the grid, the MCE trace displays the subset and join process as given below:

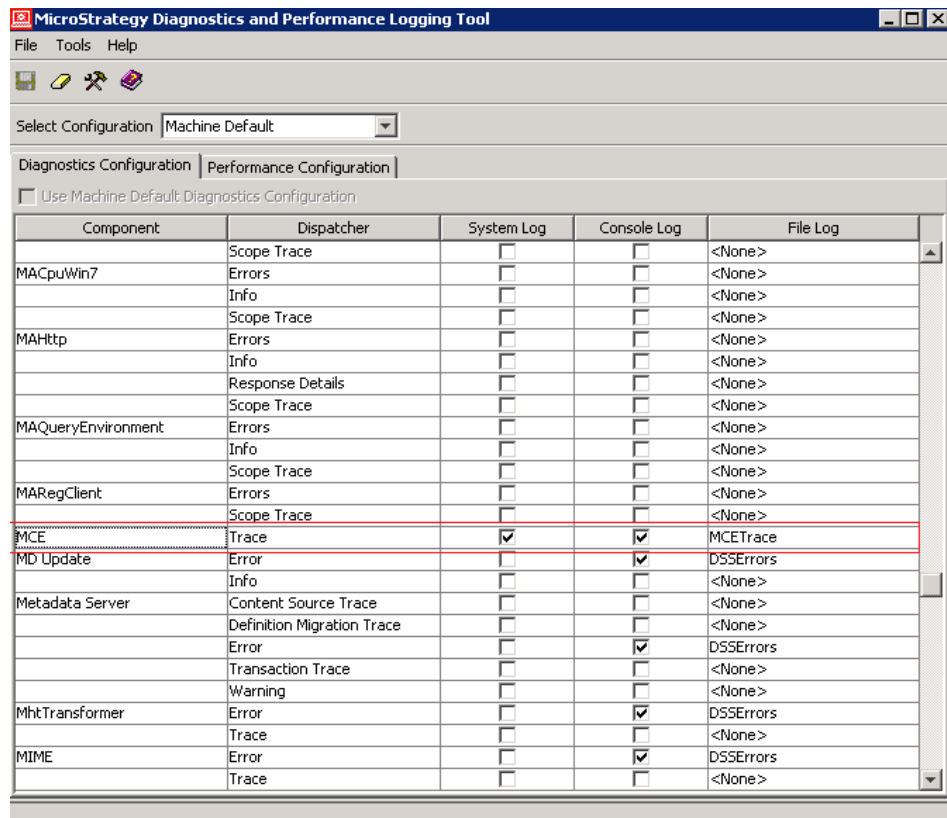
```
[PID:4740] [THR:3296] [MCE] [Trace]
select [Year]@[YEAR_ID], [Quarter]@[QUARTER_ID]
from C03 Year,Quarter to RELATION0-<Year,
Quarter>
select * from Full Outer Join(
Left Outer Join( C02 Year,Revenue, RELATION0-
<Year, Quarter> ) on Year,
Left Outer Join( C01 Quarter,Profit, RELATION0-
<Year, Quarter> ) on Quarter )
on Quarter, Year
```

Dataset Objects		?	X
▼	C01 Quarter,Profit		
▶	Quarter		
	Profit		
▼	C02 Year,Revenue		
▶	Year		
	Revenue		
▼	C03 Year,Quarter		
▶	Quarter		
▶	Year		

Quarter	Year Metrics	Revenue	Profit
2010 Q1	2010	\$8,647,238	\$297,427
2010 Q2	2010	\$8,647,238	\$294,232
2010 Q3	2010	\$8,647,238	\$354,875
2010 Q4	2010	\$8,647,238	\$357,607
2011 Q1	2011	\$11,517,606	\$440,716
2011 Q2	2011	\$11,517,606	\$402,254
2011 Q3	2011	\$11,517,606	\$462,211
2011 Q4	2011	\$11,517,606	\$434,904
2012 Q1	2012	\$14,858,864	\$549,010
2012 Q2	2012	\$14,858,864	\$512,794
2012 Q3	2012	\$14,858,864	\$571,770
2012 Q4	2012	\$14,858,864	\$615,823

How to activate the **MCE** Trace in MicroStrategy 10

- ☐ Open the **MicroStrategy Diagnostics and Performance Logging Tool** from the MicroStrategy I-Server machine.
- ☐ From the Select Configuration drop-down list, select **Machine Default** to configure logging for this machine only or for the entire server instance
- ☐ Select the **Diagnostics Configuration** tab.
- ☐ Select the component **MCE**.
- ☐ To log information about a component to the operating system log file, select the System Log check box for that component.
- ☐ To log information about a component to the MicroStrategy Monitor console, select the Console Log check box for that component.
- ☐ To log information about a component to a MicroStrategy log file, in the File Log drop-down list for that component, select the log file.
- ☐ From the File menu, select Save. **The new settings are saved in the registry.**



Demo

Is my application a good fit for PRIME?



Good Fit

All data for an application can be loaded into a single PRIME dataset

All tables can be partitioned on a single column

Centrally managed web and/or mobile dashboards with ad-hoc slicing and dicing of data

Centrally managed web and/or mobile dashboards with ad-hoc slicing and dicing of data

Derived Metrics with any Analytical Engine supported function

Not a Good Fit (**as of v10.2)

Applications that need write-back Transaction Services

Applications that require advanced ROLAP functionality, like Metric Qualifications, Report as Filter, etc.

Data that requires transformations, cleansing; Unstructured data

Summary and Takeaways

PRIME Advantages

- ✓ Larger Data Volumes
- ✓ Sub 5-second response time
- ✓ Integrated Data & Visualization Layers
- ✓ Parallel Query Execution
- ✓ Data Partitioning
- ✓ Parallel Data Loading
- ✓ Faster Cube Refreshes

Plan your Upgrade

- ✓ Configure your 9.x I-Cubes to use PRIME
- ✓ Identify a good partitioning attribute
- ✓ Enable parallel query execution
- ✓ Understand the MTDI Interface
- ✓ Decide if the application is a good fit for MTDI

Tune for Performance

- ✓ Understand CSI and MCE traces
- ✓ Apply Governing Settings
- ✓ Tune VLDB Properties
- ✓ Review Registry Settings

Additional resources

Try us now today

<http://www.microstrategy.com/us/free-trial>

Access the Community Tech Notes

TN221521: What is the Cube Subsetting Instruction (CSI) in MicroStrategy 10 ?

TN221518: How to activate the Cube Subsetting Instruction (CSI) Log in MicroStrategy 10

TN47740: How to enable MCE trace in MicroStrategy Analytical Platform 9.4.1

...and a lot more.

Join our Webcasts

<http://www.microstrategy.com/us/events/webcasts>

Q & A



Thank you.