Best Practices - Mondrian Cache Management
This page intentionally left blank.
## Contents

Overview .............................................................................................................................................................. 1

Cache Management ........................................................................................................................................... 2

  Using PDI or External Tools to Call Clear Cache with APIs ........................................................................... 2

    On API Call Details .................................................................................................................................... 2

    On Clustered Environments ..................................................................................................................... 3

  Using the Reporting/BA Scheduler .............................................................................................................. 3

    On Clustered Environments ..................................................................................................................... 3

  Using the Dynamic Schema Processor (DSP) to Segment a New Cache ................................................ 3

    On DSP Clustered Environments ............................................................................................................. 4

Related Information ........................................................................................................................................... 5
This page intentionally left blank.
Overview

This document provides methods for clearing and managing the Mondrian cache in clustered environments. The method you choose will depend on how the BA server is used and how the ETL execution processes results.

Some of the topics that will be discussed in this document are using PDI or external tools to clear the Mondrian cache with APIs, using the BA scheduler to monitor processes and manage a CLEAR-ALL-CACHE event, and using the Dynamic Schema Processor (DSP) to segment a new cache.

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pentaho</td>
<td>5.4, 6.x, 7.0</td>
</tr>
</tbody>
</table>
Cache Management

There are different cache strategies that can be used based on how the Reporting server is deployed and how the ETL execution processes results in a clustered environment.

You will only need to call one of the nodes to clear the cache; other nodes will become cleared automatically.

The following methods describe different techniques that can be used to manage the cache in Mondrian:

- Using PDI or External Tools to Call Clear Cache with APIs
- Using the BA Scheduler
- Using Dynamic Schema Processor (DSP) to Segment a New Cache

Using PDI or External Tools to Call Clear Cache with APIs

The ETL process that loads successfully modified and new data must implement a call to the BA server. For security reasons, you should apply an IP-trusted filter in order to accept requests from the calling ETL without authentication.

You need to use Analyzer to call clear cache because the API is an Analyzer API. You will only need to call one of the nodes to clear the cache; other nodes will become cleared automatically.

![Figure 1: Using PDI or External Tool to Call Clear Cache](image)

On API Call Details

You will need to have administrator permissions to run these APIs. You will receive a **TRUE** response when done successfully, and an **ERROR** response, otherwise.

1. Run this API to clear one catalog only:

   ```
   ../api/repos/xanalyzer/service/ajax/clearCache?catalog=SteelWheels&time=1464256845958
   ```

2. To clear all catalogs, run the following API:

   ```
   ../api/system/refresh/mondrianSchemaCache
   ```

The `time=<timestamp>` is not necessary in either case. It is used to force browsers or clients to submit the request and not use the cache.
**On Clustered Environments**

During distributed cache implementation, calling any of the nodes will clear the cache for the entire cluster of nodes. If no distributed cache is applied, the external process (Figure 1) will need to make the API call to the known nodes (this required maintenance of the nodes IPs) and Authentication privileges/trusted IP to the nodes.

**Using the Reporting/BA Scheduler**

This method is based on a recurrent monitoring process that can be established on a `CLEAR-ALL-CACHE` event from the API call. Scheduled Events can be logged in an ETL Status Table, Time Trigger (Fixed Time), or other.

You will only need to call one of the nodes to clear the cache; other nodes will become cleared automatically.

![Figure 2: Using the Reporting/BA Scheduler](image)

**On Clustered Environments**

During distributed cache implementation, calling any of the nodes will clear the cache for the entire cluster of nodes. If no distributed cache is implemented (Figure 2), the scheduled process will need to make the API call to the known nodes (this required maintenance of the node's IPs) and Authentication privileges/trusted IP to the nodes.

**Using the Dynamic Schema Processor (DSP) to Segment a New Cache**

This method is based on changing the Mondrian schema signature (Hash file signature) to create a new cache segment. The DSP will force the OLAP engine to segment the new cache by injecting the last successful data load `TIMESTAMP` as part of the schema definition.

You can use session or global platform variables to specify the last successful `TIMESTAMP`, depending on your use case. This new data may not be available until the new cache finishes processing.

Pentaho’s documentation has more detailed information and methods about the [Dynamic Schema Processor](#) and the [Schema Processor](#).

We recommend segmenting the cache in a controlled manner. This method is only recommended if new cache segments do not change often.
The DSP will use a `TIMESTAMP` injection in a comment on the file or in an SQL filter based on the user cases and options below.

![Diagram](image)

*Figure 3: Using DSP to Segment a New Cache*

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Adding accessible information on last update to data</strong></td>
<td>A <code>TIMESTAMP</code> is created in an accessible location for the BA server(s) after data in the fact table is changed. For example, a control table.</td>
</tr>
</tbody>
</table>
| **Reading the TIMESTAMP**                     | A session variable is loaded with the `TIMESTAMP` of the ETL status or fixed time. The name of the session variable can be schema or project, based on the use case.  
  **Note:** We recommend doing this when a user logs into any Pentaho server. |
| **Activity of the DSP**                       | Each project and schema has a DSP. The DSP will evaluate or search for specific session information (parameter-based) to modify or update the schema `TIMESTAMP` place-holder. This will force the OLAP engine to segment a new cache area. |
| **Keeping Data Integrity (optional)**         | The `TIMESTAMP` should force an SQL injection to keep user data integrity. The pattern is similar to the following:  
  ```sql  
  <SQL dialect="generic">  
  (DATE_FIELD <= TIMESTAMP) </SQL>  
  ``` |
| **Use the DSP to Inject a Condition into All Queries** | This makes sure that only the data that has been vetted and processed entirely is included in the reports that the user produces. After the ETL processes are completed, the DSP can use a new timestamp, which will instruct Mondrian to include the new data as well from that point on. |
| **Optional: Update TIMESTAMP in session or global variables** | `TIMESTAMP` can be updated in session or global variables. The schema cache will change for current logged users, in the case of global variables. |

**On DSP Clustered Environments**

This method works in *passive* mode. There is no need to do maintenance to the node's list. Each node will be responsible for making the correct schema DSP change. The DSP signature will make multiple servers use the same cache segment.
Related Information

The following links provide useful information relating to topics discussed in this document.

- Dynamic Schema Processor
- Schema Processor