



InformaticaTM

Operation

Informatica MDM - Product 360

Version: 10.5 HotFix 3 SP 1

Table of Contents

1	Database Operation	7
1.1	Database update	8
1.1.1	Prerequisites on Oracle	8
1.2	Maintenance tasks	8
1.2.1	Indices and Statistics	8
1.2.1.1	Create Maintenance Plan for SQL Server	9
1.2.1.2	Create Maintenance Plan for Oracle	14
1.2.2	Backup and Recovery	17
1.3	Tuning.....	18
1.3.1	MS-SQL Server.....	18
1.3.1.1	Activity Monitor	18
1.3.1.2	Data Collection	18
1.3.1.3	Memory Settings	19
1.3.1.4	Transaction Logs and the Recovery Mode	21
1.3.1.5	TempDB handling	21
1.4	Trouble shooting.....	21
1.4.1	Reporting.....	21
1.4.1.1	Usage statistics	21
1.4.1.2	Cleanup all temporary reports	25
1.4.1.3	Separate storage for reporting tables.....	26
1.4.2	Create missing entries Item/Product/Variant/Structure-attributes in repository key language	26
1.4.2.1	Solution	26
1.4.2.2	Consequences	27
1.4.3	MS-SQL Server.....	27
1.4.3.1	"The query processor could not start the necessary thread resources for parallel query execution"	27
1.4.3.2	"Transaction (Process ID XXX) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction"	27
1.4.4	Oracle.....	28
1.4.4.1	Import/Export of schemas.....	28
1.4.4.2	Connection Issues	28
1.4.4.3	Oracle JDBC driver OutOfMemoryError / huge memory consumption.....	29

1.4.4.4	Oracle JDBC driver OutOfMemoryError (Cache)	29
1.4.4.5	Oracle ReportStoreTemp performance	30
1.4.4.6	Automatic expiration of passwords for Oracle DB users.....	30
1.5	Diagnosing and tuning poorly performing Oracle queries	30
1.5.1	Fixing frequency histogram issues.....	37
1.6	Oracle Import/Export for MAIN, MASTER and SUPPLIER Schemas	38
1.6.1	Preparation	39
1.6.1.1	Directory Object	39
1.6.2	Export	39
1.6.2.1	User Roles	39
1.6.2.2	Execute export datapump	40
1.6.3	Import	40
1.6.3.1	User Roles	40
1.6.3.2	Tablespaces.....	40
1.6.3.3	Execute Import Datapump	41
2	Server Operation	42
2.1	Startup & Shutdown	43
2.1.1	Status Cache.....	43
2.2	Maintenance Tasks	43
2.3	Monitoring	44
2.4	Tuning.....	44
2.4.1	Memory Settings	44
2.4.2	Garbage Collection	44
2.4.2.1	GC activity log.....	45
2.4.2.2	Analyse GC activity log & optimize	45
2.4.3	Communication Framework	45
2.4.4	Mass-Data Parallel Processing	45
2.4.5	Media Asset Parallel Management	46
2.4.5.1	CreateZipManagement.....	46
2.4.5.2	MediaAssetProvider special MBean	47
2.4.5.3	HeilerClassic special MBeans	48
2.4.6	Database Connection Pool	49
2.4.7	Logging	51
2.4.8	Import	51

2.4.8.1	Preferences regarding CPU usage	51
2.4.8.2	Preferences regarding memory usage	53
2.5	Trouble Shooting	53
2.5.1	Anti-Virus Software affecting performance	53
2.5.2	Media Asset Provider (Classic Provider).....	53
2.5.2.1	No preview are visible	53
2.5.3	Obtain a Memory Dump (Heap Dump) from the Java Virtual Machine	54
2.5.4	Enhanced Logging for Assortment Content Calculation.....	54
2.5.5	Server and client in different time zones	55
2.5.6	Cluster.....	55
2.5.7	Usage of strong cryptographic algorithms to encrypt/decrypt secure information	55
2.6	DB Connection pool	56
2.6.1	High performance connection pool configuration example	56
2.6.1.1	Hibernate.....	57
2.6.1.2	UDA	58
2.7	Server Job Maintenance	60
2.7.1	Configure the life time of a job	60
2.7.2	Syntax	60
2.7.2.1	Example	60
2.7.3	Standard Job List	60
2.8	Soft Delete Cleanup Job	64
2.8.1	Why is there a concept of soft delete and a cleanup job that triggers the logging in the DB?	64
2.8.2	General information.....	65
2.8.3	Configuration	65
2.8.4	Schedule	65
2.8.5	Be cautious with mass deletes in such a setup	66
2.8.5.1	How do other customers solve such a constellation?.....	66
2.8.5.2	Possible Workarounds	66
2.9	Data Quality Operation	67
2.9.1	Logging	67
2.9.1.1	Rule engine execution.....	67
2.9.1.2	Data provisioning	67
2.9.1.3	Performance analyzing.....	68
2.9.2	Performance tuning.....	68

2.9.2.1	Cleanup of Status Entries	70
2.9.3	Issue reporting guide	71
2.9.3.1	Configuration setup	71
2.9.3.2	Information about the execution context.....	71
2.9.3.3	Logging data.....	72
2.9.3.4	Performance.....	72
2.9.3.5	Crashes	72
2.9.4	Configure rules that are affected by Dictionary Synchronization in order to increase performance of IDQ rule execution.....	72
2.9.4.1	Situation/ motivation	72
2.9.4.2	Solution	73
2.9.5	Status Cache Initialization.....	74
2.9.5.1	Overview	74
2.9.5.2	Schema Changes.....	74
2.9.5.3	Initialization	75
2.10	Check integrity of attribute names in key language	76
2.10.1	MSSQL.....	77
2.10.2	Oracle.....	77
2.11	Network Recovery	78
2.12	Monitoring with Micrometer.....	79
2.12.1	General monitoring approach	79
2.12.2	List of Product 360 custom metrics.....	79
2.12.3	Sample for monitoring a Product 360 server with Micrometer, Prometheus and Grafana	80
2.12.4	Metrics and Kibana Dashboard	82
2.12.4.1	Installation, Configuration and Operation	82
2.12.4.2	Dashboards	86
2.12.4.3	Best Practices and Recommendations	117
2.13	Trigger Backpressure	117
2.13.1	Introduction	117
2.13.2	When does Backpressure scenario happens?	118
2.13.3	What happens during backpressure?	118
2.13.4	Log Messages during Trigger Backpressure	119
2.13.5	Trigger Watchdog.....	119
2.13.6	Trigger Framework Metrics.....	121
3	Desktop Operation.....	121

3.1	Maintenance Tasks	122
3.2	Tuning.....	122
3.2.1	Performance optimization of the "Documents" view	122
3.2.1.1	Filter	123
3.2.2	Import	124
3.3	Trouble Shooting	125
3.3.1	User Interface is not responding any more (client freeze)	125
3.3.1.1	Find the problem on the client.....	125
3.3.1.2	Find the problem on the server	126
3.3.1.3	Connection timeouts between Server and Client	126
3.3.2	Rich Text Editor	127
3.3.2.1	Unable to use the clipboard actions "Cut", "Copy" and "Paste"	127
3.3.3	Workaround to prevent the "Invalid Certificate" error (Mozilla)	127
3.4	Product Paradim (EGD) limitations.....	128
4	Audit Trail Operation	129
4.1	Backup to File.....	129
4.2	Restore from File	137
5	Log File Overview	141
5.1	Additional Logging of Dataquality	142
5.2	Additional Logging of Change Summary and Datagraph	143
5.3	Additional Logging of Media Asset Processing	143

The operation manual documents actions which need to be performed in order to keep Informatica MDM - Product 360 up and running in a stable way. It contains the backup strategy as well as tuning and troubleshooting hints.

In order to meet enterprise level requirements of our current and future customers we try to steadily improve the performance of all parts of the Product 360 application. Optimal performance comes not without a cost, you will need adjust the platform configuration to the actual use case and load scenarios you experience at the customer.

1 Database Operation

Combines maintenance, tuning and troubleshooting for the Product 360 Core Database server. Additionally to the recommendations of the operation guide, a well educated database administrator should be available for maintenance and tuning tasks.

- [Database update](#) (see page 8)
 - [Prerequisites on Oracle](#) (see page 8)
- [Maintenance tasks](#) (see page 8)
 - [Indices and Statistics](#) (see page 8)
 - [Create Maintenance Plan for SQL Server](#) (see page 9)
 - [Automatic execution \(MSSQL2012\)](#) (see page 9)
 - [Manual execution](#) (see page 13)
 - [Create Maintenance Plan for Oracle](#) (see page 14)
 - [Height-Balanced Histograms on NVARCHAR2 Columns](#) (see page 15)
 - [Extended Column Statistics](#) (see page 16)
 - [Backup and Recovery](#) (see page 17)
- [Tuning](#) (see page 18)
 - [MS-SQL Server](#) (see page 18)
 - [Activity Monitor](#) (see page 18)
 - [Data Collection](#) (see page 18)
 - [Memory Settings](#) (see page 19)
 - [Lock pages in memory](#) (see page 20)
 - [Transaction Logs and the Recovery Mode](#) (see page 21)
 - [TempDB handling](#) (see page 21)
- [Trouble shooting](#) (see page 21)
 - [Reporting](#) (see page 21)
 - [Usage statistics](#) (see page 21)
 - [Cleanup all temporary reports](#) (see page 25)
 - [Separate storage for reporting tables](#) (see page 26)
 - [Create missing entries Item/Product/Variant/Structure-attributes in repository key language](#) (see page 26)
 - [Solution](#) (see page 26)
 - [Consequences](#) (see page 27)
 - [MS-SQL Server](#) (see page 27)
 - ["The query processor could not start the necessary thread resources for parallel query execution"](#) (see page 27)
 - ["Transaction \(Process ID XXX\) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction"](#) (see page 27)
 - [Oracle](#) (see page 28)

- [Import/Export of schemas](#) (see page 28)
- [Connection Issues](#) (see page 28)
- [Oracle JDBC driver OutOfMemoryError / huge memory consumption](#) (see page 29)
- [Oracle JDBC driver OutOfMemoryError \(Cache\)](#) (see page 29)
- [Oracle ReportStoreTemp performance](#) (see page 30)
- [Automatic expiration of passwords for Oracle DB users](#) (see page 30)

1.1 Database update

1.1.1 Prerequisites on Oracle

In case the Product 360 database is running in archive log mode (most likely), make sure to turn archive log mode off before executing the database update setup.

Otherwise you might risk exceeding available archive log disk space, in which case any database activity will be suspended immediately, and it will stay suspended until there is free available disk space again.

1.2 Maintenance tasks

1.2.1 Indices and Statistics

The database uses indices to increase the performance for read access on columns. In Product 360 we have indices on the columns which are typically used for selecting data, for example, all logical keys are at least part of an index. Additionally to that, the database uses statistics about the data in the tables in order to choose the most effective execution plan. Indices as well as statistics need to be maintained by the database administrator on a regular basis. Usually this is being done by creating a Maintenance Task which executes every day.

However, the typical maintenance task is more something of a bulldozer in maintaining. This means, it just rebuilds or reorganizes all indices no matter if needed or not and it also kills all statistics and rebuilds them no matter if needed or good for the statistic. Sometimes statistics which are very accurate get blindly rebuild and the result is a statistic which is not that accurate anymore, leading to a lesser performance than what you could have.

So, the general recommendation (from Microsoft Consulting) is like this (pseudo code):

```

if (number of pages in index < 1000)
  do nothing
else
  if (fragmentation < 5%)
    do nothing
  else if (fragmentation between 5% and < 30%)

```



```
reorganize index + update its statistics
```

```
else
```

```
rebuild index
```

Additionally to that you should update statistics for all non-indexed columns and monitor index fragmentation to adjust the fill factor.

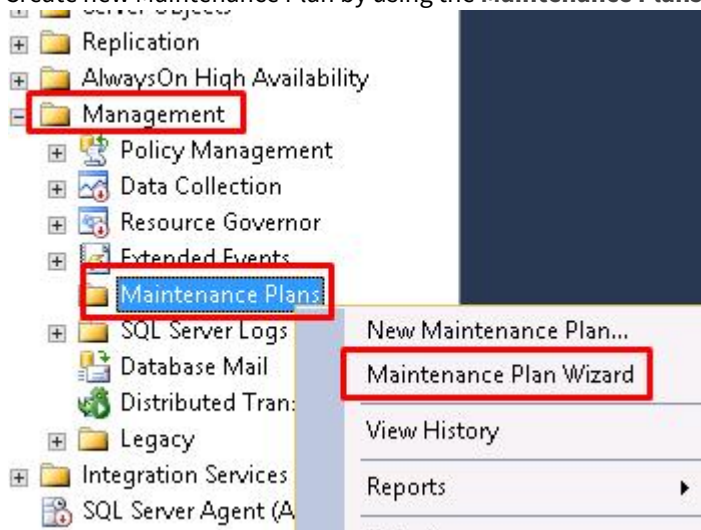
1.2.1.1 Create Maintenance Plan for SQL Server

All these steps can be combined in a single, easy to execute stored procedure which has generously been provided by Ola Hallengren. For a detailed description about the index and statistic maintenance script please refer to the online description which is available here:

<http://ola.hallengren.com/sql-server-index-and-statistics-maintenance.html>. It will create all necessary objects for you. (For your convenience we attached the script it also to this wiki page)

Automatic execution (MSSQL2012)

1. **Execute** the MaintenanceSolution.sql script on your database. Be sure that your **SQL Server Agent** (service) is running.
2. Create new Maintenance Plan by using the **Maintenance Plans Wizard** as shown below



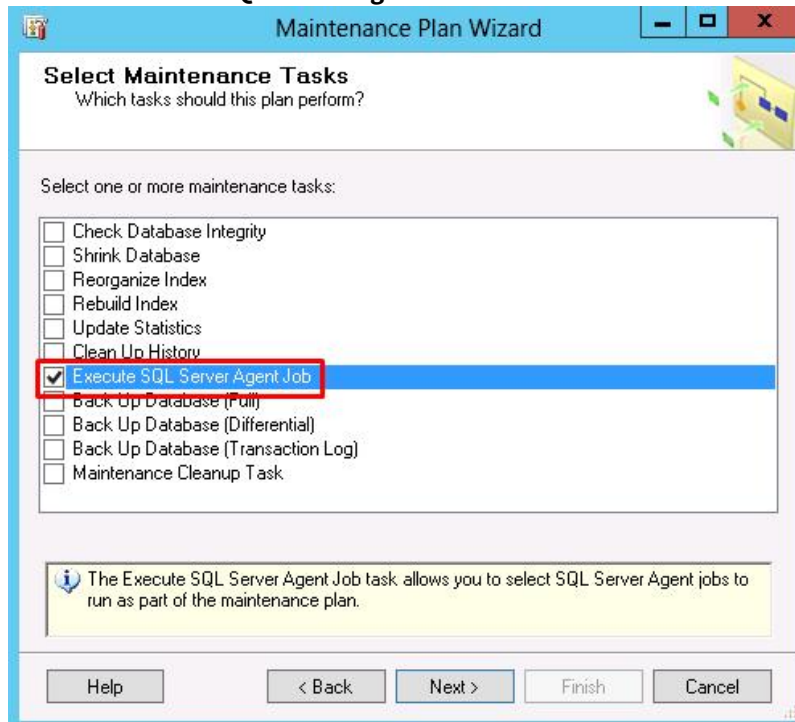
In case you get following error:



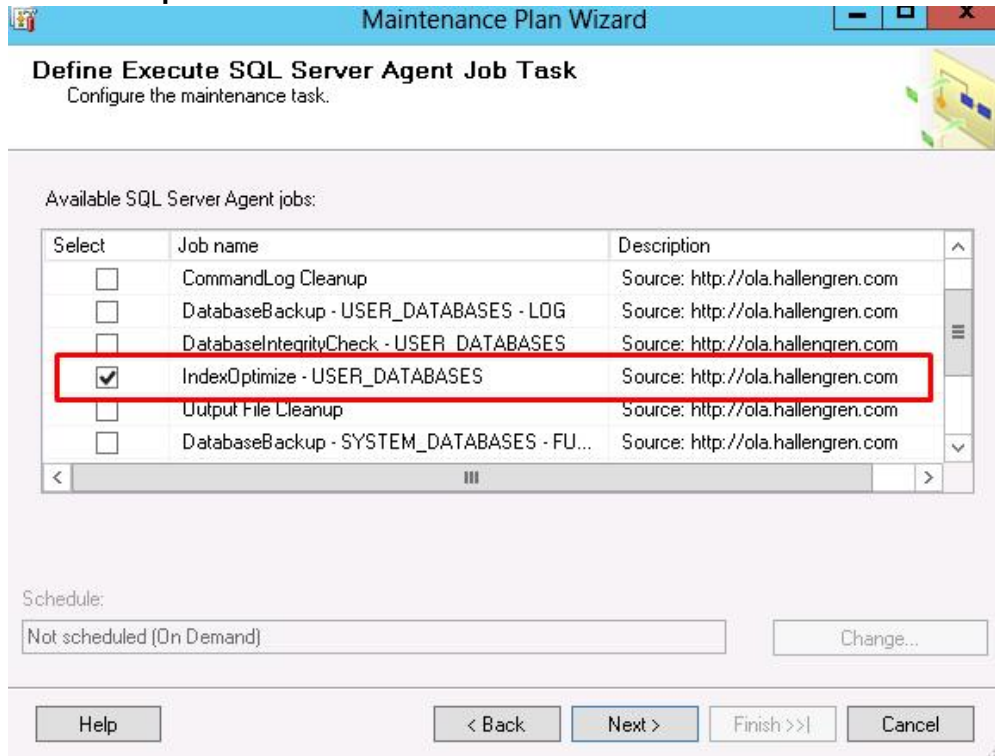
execute this statement:

```
sp_configure 'show advanced options', 1;
GO
RECONFIGURE;
GO
sp_configure 'Agent XPs', 1;
GO
RECONFIGURE
GO
```

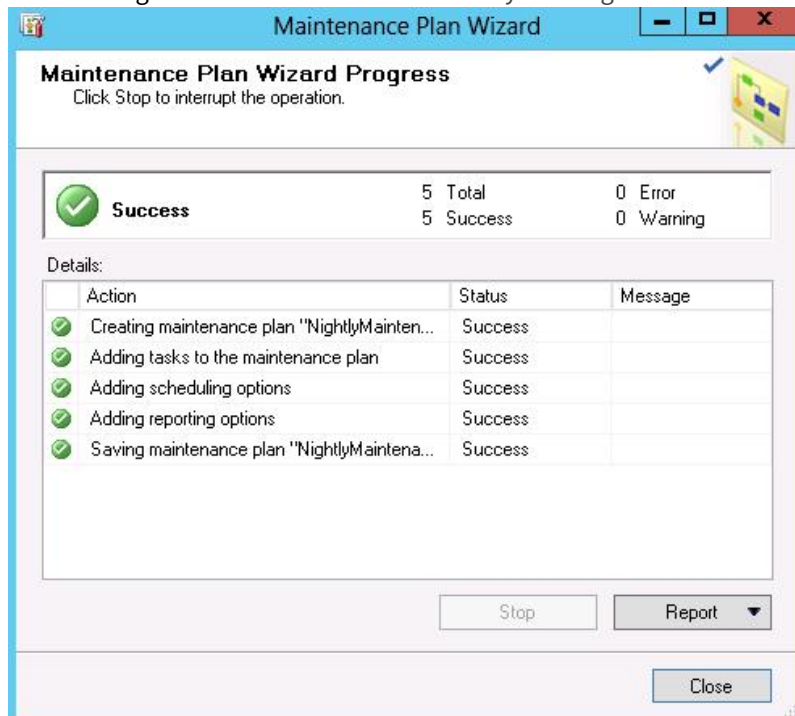
3. Type the name of your new maintenance plan and configure your job schedule, then click on **Next**
4. Select the **Execute SQL Server Agent Job** and click on **Next twice**



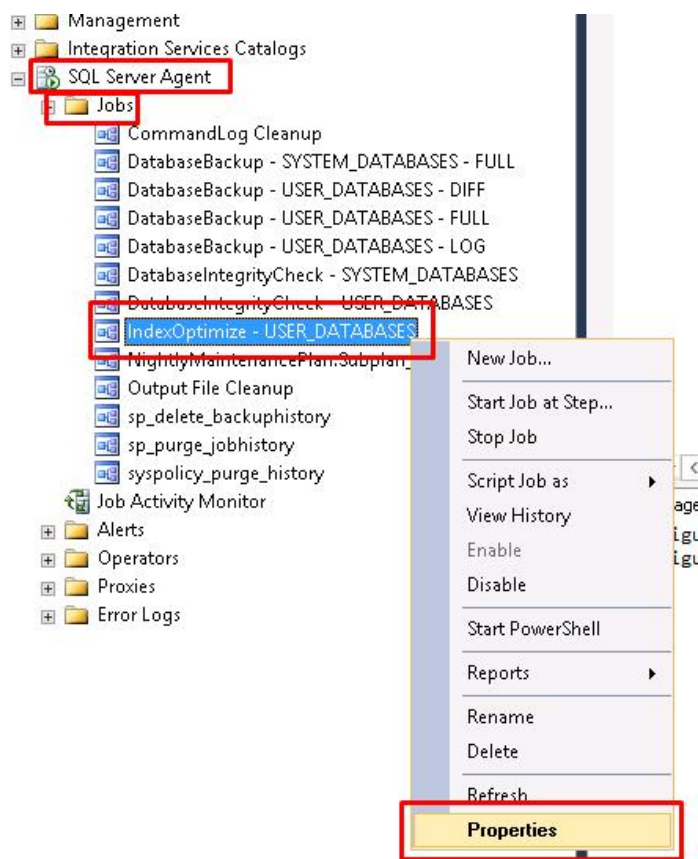
5. Choose **IndexOptimize** and click on **Next**



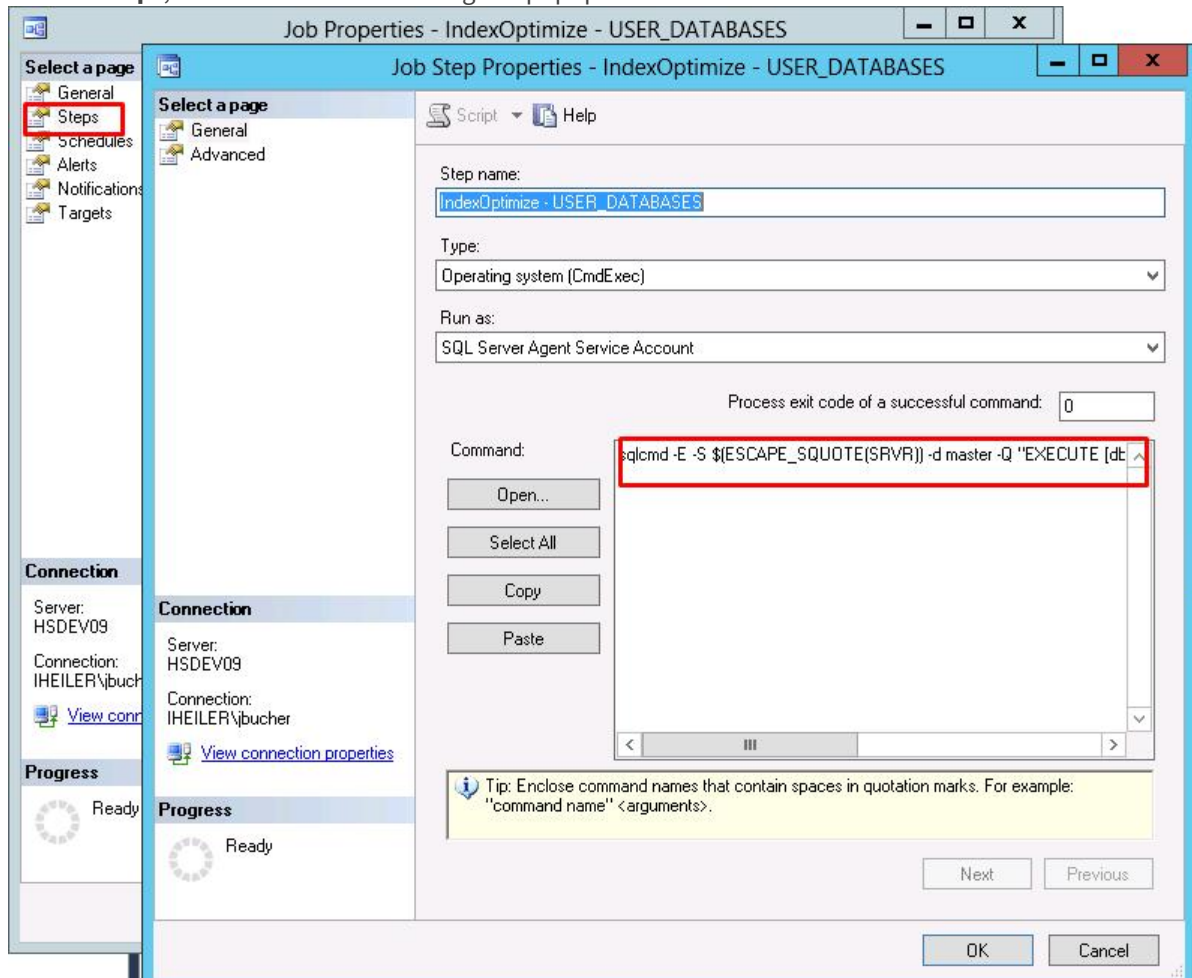
6. (Optional) Configure the report location and click on Next
7. After clicking **Finish** the task will check for any warnings or errors:



8. Click on the **SQL Server Agent** node from the left-hand side to expand the menu and open the **Jobs** node. Then right-click on **IndexOptimize** and go to the **Properties** view as shown below:



9. Click on **Steps**, then **Edit** and a new Dialog will popup.



10. **Modify** the command statement (in this example the job will be executed on all user databases)

```
sqlcmd -E -S $(ESCAPE_SQUOTE(SRVR)) -d master -Q "EXECUTE [dbo].[IndexOptimize]
@Databases = 'USER_DATABASES', @FragmentationLow = NULL, @FragmentationMedium =
'INDEX_REORGANIZE,INDEX_REBUILD_ONLINE,INDEX_REBUILD_OFFLINE',
@FragmentationHigh = 'INDEX_REBUILD_ONLINE,INDEX_REBUILD_OFFLINE',
@FragmentationLevel1 = 5, @FragmentationLevel2 = 30, @UpdateStatistics =
'COLUMNS', @OnlyModifiedStatistics = 'Y', @LogToTable = 'Y'" -b
```

Make sure to reformat above statement to a valid format.

11. Click **OK**
12. Schedule the job in the **Properties** view

Manual execution

1. Download and **execute** the MaintenanceSolution.sql script.
2. **Execute** following SQL statement (in this example the index will get optimized on PIM_MAIN, PIM_MASTER and PIM_SUPPLIER)

SQL Statement to execute

```
EXECUTE dbo.IndexOptimize @Databases = 'PIM_MAIN, PIM_MASTER, PIM_SUPPLIER',
@FragmentationLow = NULL, @FragmentationMedium =
'INDEX_REORGANIZE,INDEX_REBUILD_ONLINE,INDEX_REBUILD_OFFLINE',
@FragmentationHigh = 'INDEX_REBUILD_ONLINE,INDEX_REBUILD_OFFLINE',
@FragmentationLevel1 = 5, @FragmentationLevel2 = 30, @UpdateStatistics =
'COLUMNS', @OnlyModifiedStatistics = 'Y'
```

Make sure to reformat above statement to a valid format.

1.2.1.2 Create Maintenance Plan for Oracle

If the Oracle database instance is fully dedicated to PIM, it is recommended to disable the system job which creates plan directives because these directives can cause performance issues.

Disable Automated Statistics Collection Job

```
BEGIN
DBMS_AUTO_TASK_ADMIN.DISABLE
(
  CLIENT_NAME => 'sql tuning advisor',
  OPERATION => NULL,
  WINDOW_NAME => NULL
);
END;
/
```

If the Oracle database instance is fully dedicated to PIM, it is recommended to disable the system job which automatically updates statistics on a daily basis because it does not create histograms:

Disable Automated Statistics Collection Job

```
BEGIN
DBMS_AUTO_TASK_ADMIN.DISABLE
(
  CLIENT_NAME => 'auto optimizer stats collection',
  OPERATION => NULL,
  WINDOW_NAME => NULL
);
END;
/
```

Instead, the DBA should create its own job for statistics maintenance. It should be executed on a periodic basis, recommended nightly but also ad-hoc after more than 20% of the data has been significantly changed (e.g. during a large import).

Here is an example script which covers default Product 360 installations, the DBA is encouraged to adjust it accordingly to the specific Product 360 environment:

Gather Statistics

```
--
*****
-- Statistics collection with SKEWONLY
-- * Automatically creates histograms on any column that shows a skew in data
distribution
-- * Despite time consumption it is always recommended to use ESTIMATE_PERCENT = 100
--
*****
BEGIN
  DBMS_STATS.GATHER_SCHEMA_STATS(
    OWNNAME      => 'PIM_MAIN',
    OPTIONS       => 'gather',
    ESTIMATE_PERCENT => 100,
    METHOD_OPT     => 'for all columns size skewonly',
    CASCADE       => true,
    DEGREE        => 8
  );
  DBMS_STATS.GATHER_SCHEMA_STATS(
    OWNNAME      => 'PIM_MASTER',
    OPTIONS       => 'gather',
    ESTIMATE_PERCENT => 100,
    METHOD_OPT     => 'for all columns size skewonly',
    CASCADE       => true,
    DEGREE        => 8
  );
  DBMS_STATS.GATHER_SCHEMA_STATS(
    OWNNAME      => 'PIM_SUPPLIER',
    OPTIONS       => 'gather',
    ESTIMATE_PERCENT => 100,
    METHOD_OPT     => 'for all columns size skewonly',
    CASCADE       => true,
    DEGREE        => 8
  );
END;
/
```

Height-Balanced Histograms on NVARCHAR2 Columns

In case there are more than 254 distinct values, Oracle can't use a frequency histogram anymore but switches to a height-balanced histogram instead.

There are known performance issues with height-balanced histograms on NVARCHAR2 columns and it is recommended to delete all of these height-balanced histograms (otherwise the optimizer might create bad execution plans due to unpopular / non-existent value choices).

Delete Height-Balanced Histograms for all NVARCHAR2 Columns

```

SET SERVEROUTPUT ON
BEGIN
  FOR COL_ITEM IN (
    SELECT stats.OWNER, stats.TABLE_NAME, stats.COLUMN_NAME
    FROM DBA_TAB_COL_STATISTICS stats
    INNER JOIN DBA_TAB_COLUMNS cols
    ON stats.TABLE_NAME = cols.TABLE_NAME
    AND stats.COLUMN_NAME = cols.COLUMN_NAME
    WHERE cols.DATA_TYPE = 'NVARCHAR2'
    AND stats.HISTOGRAM <> 'NONE'
    AND stats.NUM_DISTINCT > 254
    AND stats.OWNER = cols.OWNER
    AND stats.OWNER IN ('PIM_MAIN','PIM_MASTER','PIM_SUPPLIER')
    ORDER BY stats.OWNER, stats.TABLE_NAME, stats.COLUMN_NAME
  )
  LOOP
    DBMS_OUTPUT.PUT_LINE( 'Deleting height-balanced histogram for: ' ||
COL_ITEM.OWNER || '."' || COL_ITEM.TABLE_NAME || '."' || COL_ITEM.COLUMN_NAME ||
' "' );
    DBMS_STATS.DELETE_COLUMN_STATS (OWNNAME=>COL_ITEM.OWNER, TABNAME=>'"' ||
COL_ITEM.TABLE_NAME || '"', COLNAME=>'"' || COL_ITEM.COLUMN_NAME || '"',
COL_STAT_TYPE=>'HISTOGRAM');
  END LOOP;
END;
/

```

Extended Column Statistics



In case you are running Oracle 12.1 be aware of the "extended column statistics" feature. We strongly recommend turning off this feature, and to make sure that there aren't any extended statistics columns in all PIM schemas.

The following script can be used to delete system generated extended statistics:

Delete Extended Column Statistics

```

SET SERVEROUTPUT ON
BEGIN
  FOR c IN
  (
    SELECT OWNER, TABLE_NAME, DBMS_LOB.SUBSTR(EXTENSION, 3000) X
    FROM DBA_STAT_EXTENSIONS
    WHERE OWNER LIKE 'PIM_%'
    AND CREATOR = 'SYSTEM'
    AND DROPPABLE = 'YES'
  )

```



```

        ORDER BY TABLE_NAME, X
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE('Deleting system generated column stats for "' ||
c.OWNER || '".' || c.TABLE_NAME || '" : ' || c.X);
        DBMS_STATS.DROP_EXTENDED_STATS(c.OWNER, '"' || c.TABLE_NAME || '"', c.X );
    END LOOP;
END;
/

```

It is recommended to disable the extended column statistics feature.

Disable Extended Column Statistics

```
ALTER SYSTEM SET "_optimizer_enable_extended_stats"=FALSE SCOPE=BOTH;
```

If they can't be disabled entirely, the following script can be used to disable extended column statistics only for all PIM schemas (possible starting with 12.2):

Disable Extended Column Statistics (>= 12.2)

```

BEGIN
    FOR TABLE_ITEM IN (
        SELECT t.OWNER, t.TABLE_NAME
        FROM DBA_TABLES t
        WHERE t.OWNER IN ('PIM_MAIN', 'PIM_MASTER', 'PIM_SUPPLIER')
        ORDER BY t.OWNER, t.TABLE_NAME
    )
    LOOP
        DBMS_STATS.SET_TABLE_PREFS(TABLE_ITEM.OWNER, '"' || TABLE_ITEM.TABLE_NAME || '"',
'AUTO_STAT_EXTENSIONS', 'OFF');
    END LOOP;
END;
/

```

1.2.2 Backup and Recovery

Usually the backup and recovery strategy is defined by the customers IT department and it's dba. Product 360 core works fine with nightly backups using the "simple" recovery mode of the database, as well as transaction log backups during the day using the "full" recovery mode. Choose whatever your customer required in order to feel safe.



If you switch the schemas to recovery mode "full" you have to set up the transaction log backups. If you don't your transaction logs will keep on growing eventually without limits!

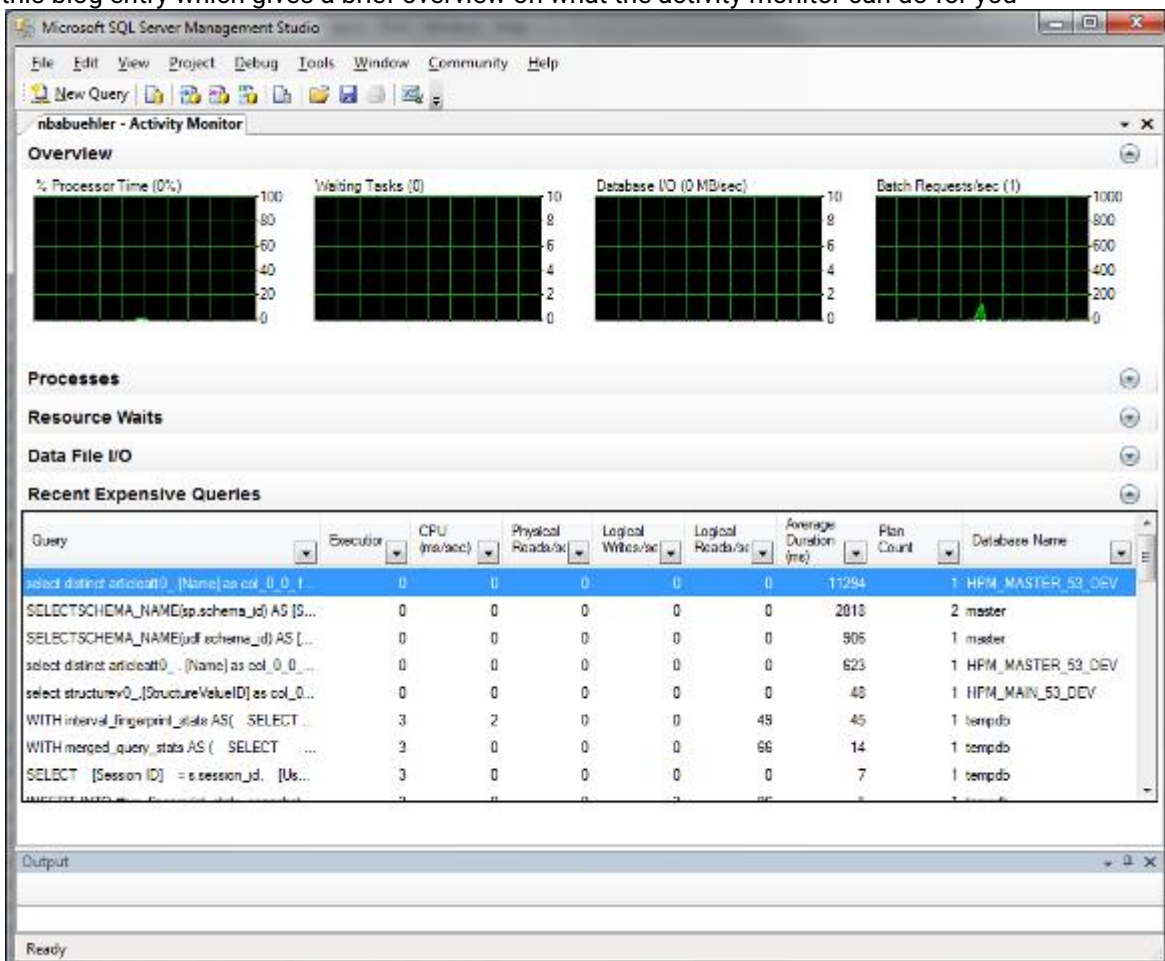
The backup plan should include the MASTER, SUPPLIER and MAIN schemas as well as all the runtime folders like import, export where the import and export files are stored.

1.3 Tuning

1.3.1 MS-SQL Server

1.3.1.1 Activity Monitor

Use the Activity Monitor which is available in SQL Server to quickly identify slow statements and connection problems. Please see this knowledge base article which explains how to open the activity monitor, as well as this blog entry which gives a brief overview on what the activity monitor can do for you



1.3.1.2 Data Collection

The activity monitor is a quite effective tool to get an overview of the current load on the database. However, sometime it's not that easy to produce the load on the application right at the time you're looking at the activity monitor. Sql Server 2008 has a new feature called "Data Collection". It might be a good idea to setup

this feature and collect performance data for some days or weeks in order to see trends.
For a detailed description as how to setup this feature please refer to the MS-SQL Server Books
Online: <http://msdn.microsoft.com/en-us/library/bb677356.aspx>



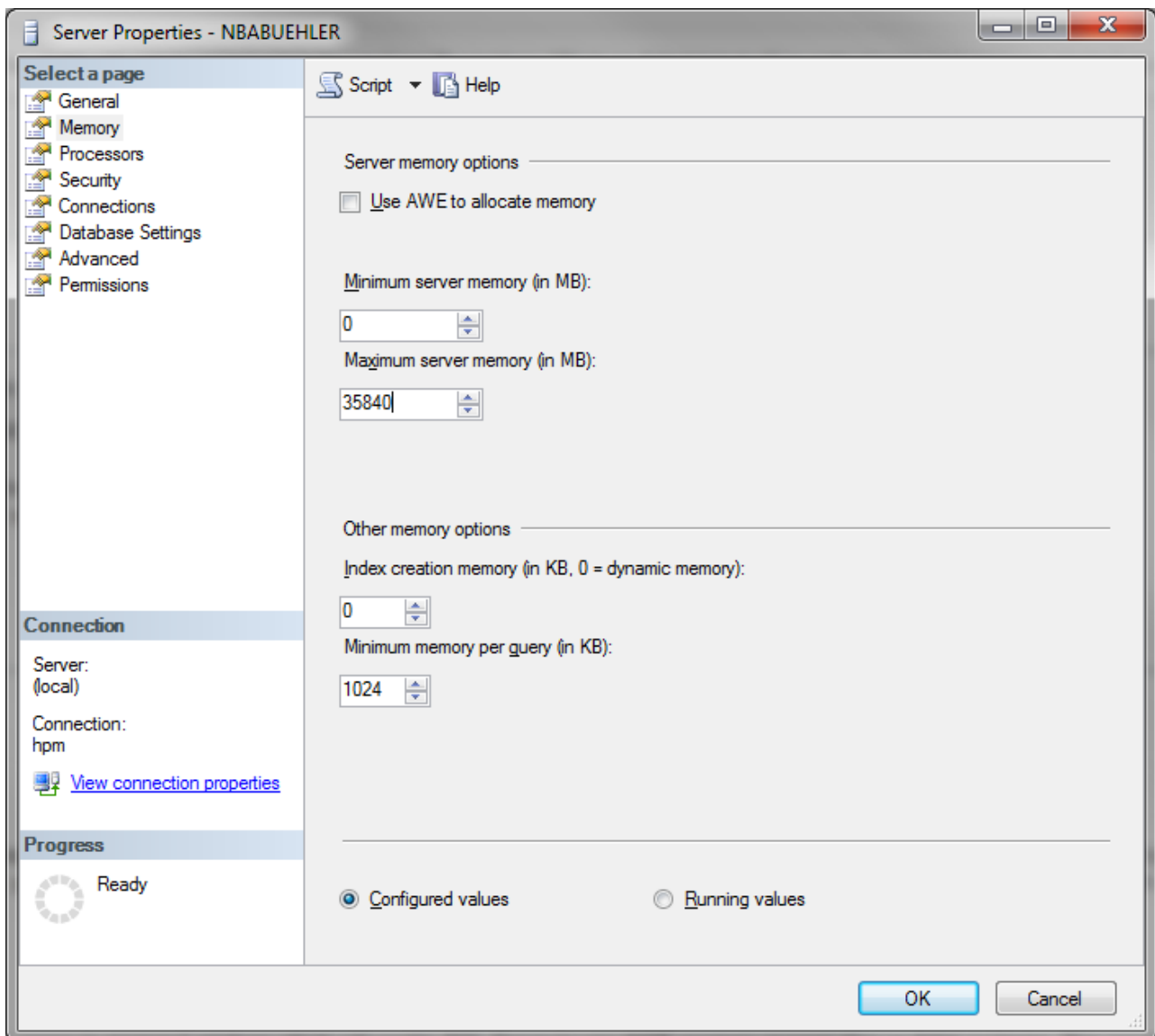
Please note that the data collection feature and the jobs it creates can not be deleted again, they can only be "disabled" so they will no longer collect any data.

1.3.1.3 Memory Settings

You should always limit the maximum memory usage of the SQL Server. In case you have a single instance on the machine, you can limit the memory to the maximum physical memory minus one GB for the operating system.

For example: The machine has 36 GB of memory, you would configure the SQL Server to use $35 \times 1024 = 35840$ MB memory, leaving the operating system 1024 MB - which should be enough, usually.

You can configure this using the Microsoft Sql Server Management Studio. Open the properties dialog of your database and adjust the memory settings like shown below.



The "Use AWE to allocate memory" will have no effect in 64bit installations, SQL Server uses AWE access automatically when the user has "Lock pages in memory" privilege.

Lock pages in memory

Enable the SQL Server Service user with the "Lock pages in memory" privilege which will enable SQL Server to use AWE memory access also on 64bit systems. This results in a better performance for the memory access since SQL Server can directly access the memory and must not ask for it using the Virtual Memory Manager (VMM).

See also "Configuring the Server for Optimal Performance" (SQL Server 2008 Administration from Wrox Press, Chapter 11, Page 434)

1.3.1.4 Transaction Logs and the Recovery Mode

In a productive environment you should define the initial size of the database log files to the expected maximum.

For both data and log files we recommend setting a proper growth value because the default of 64MB is too small.



A data base growth action always "stops the world" of the database until the files are enlarged. When this will occur often it will be a serious performance problem.

In case you are operating the Product 360 databases in **FULL** recovery mode please make sure that you take backups of the transaction logs on a regular basis.

Otherwise the transaction logs will grow endlessly because SQL Server will not re-use the existing space until a backup was made.

1.3.1.5 TempDB handling

The TempDB will be used a lot since Product 360 uses transaction mode **READ_COMMITTED_SNAPSHOT** starting with PIM 7.1.

We recommend to tune the storage and the amount of data files according to the database load.

1.4 Trouble shooting

1.4.1 Reporting

A key functionality of the Product 360 - Server is the so called "reporting". A report in our sense is more or less only a list of internal database ID's. The report therefore reflects an arbitrary amount of "objects" in Product 360 - Server, identified by their ID. Reporting is a crucial part of Product 360 - Server since many functions rely on reports. For example the table based views of root entities like items or products, the export and the delete operations as well.

1.4.1.1 Usage statistics

The Informatica Software Support might ask you to provide usage statistics for the reporting in case we need to trouble shoot an issue of yours, for these you need to execute the following script on each data source (main, master and supplier):

Usage Statistic MSSQL

```
1 CREATE TABLE [#ReportInstances]
2 (
3     [ReportInstance] NVARCHAR(30)
```

```

4      )
5      INSERT INTO [#ReportInstances] ( [ReportInstance] ) VALUES (N'ReportStore')
6      , (N'ReportStoreTempA'), (N'ReportStoreTempB')
7
8      DECLARE @REPORT_TABLE_NAME NVARCHAR(30)
9      DECLARE @ReportInstance NVARCHAR(30)
10     DECLARE @SQL_STRING NVARCHAR(MAX)
11     DECLARE @TMP_SQL NVARCHAR(MAX)
12
13     DECLARE C0 CURSOR FOR
14     SELECT [ReportInstance]
15     FROM [#ReportInstances]
16     ORDER BY [ReportInstance]
17     OPEN C0
18     FETCH NEXT FROM C0 INTO @ReportInstance
19     WHILE @@FETCH_STATUS = 0
20     BEGIN
21         SET @SQL_STRING = ''
22         DECLARE C1 CURSOR FOR
23         SELECT TABLE_NAME
24         FROM INFORMATION_SCHEMA.TABLES
25         WHERE TABLE_NAME LIKE @ReportInstance + '[0-9]%'
26         ORDER BY CAST(REPLACE(TABLE_NAME, @ReportInstance, '') AS INT)
27         OPEN C1
28         FETCH NEXT FROM C1 INTO @REPORT_TABLE_NAME
29         WHILE @@FETCH_STATUS = 0
30         BEGIN
31             IF LEN(@SQL_STRING) > 0
32             BEGIN
33                 SET @SQL_STRING = CONCAT(@SQL_STRING, ' UNION ALL ')
34             END
35             SET @SQL_STRING = CONCAT(@SQL_STRING, 'SELECT * FROM dbo.[' +
36             @REPORT_TABLE_NAME + '])')
37             FETCH NEXT FROM C1 INTO @REPORT_TABLE_NAME
38         END
39         CLOSE C1
40         DEALLOCATE C1
41
42         SET @TMP_SQL = CONCAT('SELECT COUNT(*) AS [Number of Report IDs from ',
43         @ReportInstance, '] FROM (', @SQL_STRING, ') X')
44         PRINT @TMP_SQL
45         EXECUTE sp_executesql @TMP_SQL
46
47         SET @TMP_SQL = CONCAT('SELECT COUNT(*) AS [Number of Temporary Report
48         IDs from ', @ReportInstance, '] FROM (', @SQL_STRING, ') X WHERE ReportID
49         IN (SELECT ID FROM Report WHERE Purpose = 1)')
50         PRINT @TMP_SQL
51         EXECUTE sp_executesql @TMP_SQL
52
53         SET @TMP_SQL = CONCAT('SELECT COUNT(*) AS [Number of Permanent Report
54         IDs from ', @ReportInstance, '] FROM (', @SQL_STRING, ') X WHERE ReportID
55         IN (SELECT ID FROM Report WHERE Purpose = 3)')
56         PRINT @TMP_SQL

```

```

50      EXECUTE sp_executesql @TMP_SQL
51
52      SET @TMP_SQL = CONCAT('SELECT COUNT(*) AS [Number of Orphan Report IDs
from ', @ReportInstance, ']' FROM (' , @SQL_STRING, ') X WHERE ReportID NOT
IN (SELECT ID FROM Report)')
53      PRINT @TMP_SQL
54      EXECUTE sp_executesql @TMP_SQL
55
56      FETCH NEXT FROM C0 INTO @ReportInstance
57  END
58  CLOSE C0
59  DEALLOCATE C0
60
61  DROP TABLE [#ReportInstances]
62
63  SELECT COUNT([ID]) AS repTotalCount, Purpose, SUM([Count]) totalIdsCount
FROM [dbo].[Report] GROUP BY Purpose
64
65  SELECT count([ID]) AS numOfReports, [Count] AS repSize, COUNT([ID])*[Count]
AS totalSize FROM [dbo].[Report] WHERE Purpose=1 GROUP BY [Count] ORDER
BY totalSize DESC
66  SELECT count([ID]) AS numOfReports, [Count] AS repSize, COUNT([ID])*[Count]
AS totalSize FROM [dbo].[Report] WHERE Purpose=1 GROUP BY [Count] ORDER
BY numOfReports DESC
67  SELECT count([ID]) AS numOfReports, [Count] AS repSize, COUNT([ID])*[Count]
AS totalSize FROM [dbo].[Report] WHERE Purpose=1 GROUP BY [Count] ORDER
BY repSize DESC
68
69  SELECT [Type], SUM([Count]) AS totalSize FROM [dbo].[Report] WHERE
Purpose=1 AND [Count] < 10 GROUP BY [Type]
70  SELECT [Type], SUM([Count]) AS totalSize FROM [dbo].[Report] WHERE
Purpose=1 AND [Count] < 100 GROUP BY [Type]
71  SELECT [Type], SUM([Count]) AS totalSize FROM [dbo].[Report] WHERE
Purpose=1 AND [Count] < 200 GROUP BY [Type]
72
73  SELECT TOP 100 [ID] ,[Type] ,[Purpose] ,[Count] ,[CreationDate] FROM
[dbo].[Report] WHERE Purpose=1 ORDER BY CreationDate DESC

```

Usage Statistic Oracle

```

1  SET SERVEROUTPUT ON
2  SET LINESIZE 2000
3
4  DECLARE
5      REPORT_TABLE_NAME VARCHAR2(30);
6      SQL_STRING VARCHAR2(2000);
7      TMP_SQL VARCHAR2(2000);
8      TMP_RESULT NUMBER;
9      TEMP_NAME NVARCHAR2(30);
10     TYPE TEMP_ARRAY IS VARRAY(3) OF VARCHAR2(30);

```

```

11  ARRAY TEMP_ARRAY := TEMP_ARRAY('ReportStore', 'ReportStoreTempA',
12  'ReportStoreTempB');
13
14  CURSOR C1(REPORT_INSTANCE VARCHAR2)
15      IS SELECT TABLE_NAME
16      FROM USER_TABLES
17      WHERE REGEXP_LIKE(TABLE_NAME, '^' || REPORT_INSTANCE ||
18  '\d')
19      ORDER BY CAST(REPLACE(TABLE_NAME, REPORT_INSTANCE) AS
20  NUMBER);
21  BEGIN
22
23  FOR i IN 1..ARRAY.COUNT LOOP
24      TEMP_NAME := ARRAY(i);
25      SQL_STRING := '';
26
27      OPEN C1(TEMP_NAME);
28      LOOP
29          FETCH C1 INTO REPORT_TABLE_NAME;
30          EXIT WHEN C1%NOTFOUND;
31
32          IF LENGTH(SQL_STRING) > 0 THEN
33              BEGIN
34                  SQL_STRING := SQL_STRING || ' UNION ALL ';
35              END;
36          END IF;
37
38          SQL_STRING := SQL_STRING || ' SELECT * FROM "' || REPORT_TABLE_NAME
39  || '"';
40      END LOOP;
41      CLOSE C1;
42
43      TMP_SQL := 'SELECT COUNT(*) AS "numIds' || TEMP_NAME || '" FROM (' ||
44  SQL_STRING || ') X';
45      DBMS_OUTPUT.PUT_LINE(TMP_SQL);
46      EXECUTE IMMEDIATE(TMP_SQL) INTO TMP_RESULT;
47      DBMS_OUTPUT.PUT_LINE(TMP_RESULT || CHR(10));
48
49      TMP_SQL := 'SELECT COUNT(*) AS "numTempIds' || TEMP_NAME || '" FROM ('
50  || SQL_STRING || ') X WHERE "ReportID" IN (SELECT "ID" FROM "Report"
51  WHERE "Purpose" = 1)';
52      DBMS_OUTPUT.PUT_LINE(TMP_SQL);
53      EXECUTE IMMEDIATE(TMP_SQL) INTO TMP_RESULT;
54      DBMS_OUTPUT.PUT_LINE(TMP_RESULT || CHR(10));
55
56      TMP_SQL := 'SELECT COUNT(*) AS "numPermIds' || TEMP_NAME || '" FROM ('
57  || SQL_STRING || ') X WHERE "ReportID" IN (SELECT "ID" FROM "Report"
58  WHERE "Purpose" = 3)';
59      DBMS_OUTPUT.PUT_LINE(TMP_SQL);
60      EXECUTE IMMEDIATE(TMP_SQL) INTO TMP_RESULT;
61      DBMS_OUTPUT.PUT_LINE(TMP_RESULT || CHR(10));

```



```

53      TMP_SQL := 'SELECT COUNT(*) AS "numOprhanIds' || TEMP_NAME || ' " FROM
(' || SQL_STRING || ') X WHERE "ReportID" NOT IN (SELECT "ID" FROM
"Report")';
54      DBMS_OUTPUT.PUT_LINE(TMP_SQL);
55      EXECUTE IMMEDIATE(TMP_SQL) INTO TMP_RESULT;
56      DBMS_OUTPUT.PUT_LINE(TMP_RESULT || CHR(10));
57
58  END LOOP;
59  END;
60  /
61
62  SELECT COUNT ("ID") AS "repTotalCount", "Purpose", SUM ("Count") AS
"totalIdsCount" FROM "Report" GROUP BY "Purpose";
63  SELECT COUNT ("ID") AS "numOfReports", "Count" AS "repSize", COUNT ("ID")*"
Count" AS "totalSize" FROM "Report" WHERE "Purpose"=1 GROUP BY "Count"
ORDER BY "totalSize" DESC;
64  SELECT COUNT ("ID") AS "numOfReports", "Count" AS "repSize", COUNT ("ID")*"
Count" AS "totalSize" FROM "Report" WHERE "Purpose"=1 GROUP BY "Count"
ORDER BY "repSize" DESC;
65
66  SELECT "Type", SUM ("Count") AS "totalSize" FROM "Report" WHERE "Purpose"=
1 AND "Count" < 10 GROUP BY "Type";
67  SELECT "Type", SUM ("Count") AS "totalSize" FROM "Report" WHERE "Purpose"=
1 AND "Count" < 100 GROUP BY "Type";
68
69  SELECT "Type", SUM ("Count") AS "totalSize" FROM "Report" WHERE "Purpose"=
1 AND "Count" < 200 GROUP BY "Type";
70  SELECT "ID", "Type", "Purpose", "Count", "CreationDate" FROM "Report"
WHERE "Purpose" = 1 ORDER BY "CreationDate" DESC;

```

1.4.1.2 Cleanup all temporary reports

The Product Manager uses an automatically scheduled system job for to delete temporary reports which are older than one day. This job is scheduled every hour (by default), therefore the `Report` table should not have any entry with `Purpose=1` (temporary) which are older than 24 hours. However, it occurred in the past that this job could not fulfill its task in a reasonable amount of time or due some misconfiguration.

Prior to Version 6.0 this job deleted all reports (and their corresponding entries in the `ReportStoreTemp` table) in a **single transaction**. This may lead to an "explosion" of the transaction log since the `ReportStoreTemp` table might contain millions of items. In case you find old reports in the `Report` table (in any Product 360 - Server schema) you can execute the following SQL statements to completely delete all temporary reports in one small step, keeping the transaction log usage low.

Please note that you should shutdown the Product 360 - Server during this - since currently running operations might rely on a temporary report.

Delete ALL Temporary Reports

```
1  DELETE FROM [Report] WHERE [Purpose] = 1
```

With PIM - Server versions ≥ 6.0 the delete job is more aware of this problem. Big reports will be deleted one by one, small reports in a bulk delete operation. You can configure the boundaries of "big reports" in the `plugin_customization.ini` file. Additionally to that, the delete job can also be configured to delete reports which are younger than a day - for example a few hours.

1.4.1.3 Separate storage for reporting tables

On heavy load databases we recommend to put the corresponding reporting tables for each schema:

- Report
- ReportStore
- ReportStoreTempA
- ReportStoreTempB

on a high-performance storage separated from the rest of the Product 360 tables.

1.4.2 Create missing entries Item/Product/Variant/Structure-attributes in repository key language

This can happen if the language of the repository got changed during maintenance, if there is any custom code in the system which adds entries by not honoring the key language or an import which is ignoring the fact the the key language should be filled always. The repository language is configured in the `server.properties` and should get never touched in a running system.

Since PIM 7.0.01 the Product 360 Core will check this behavior during server startup. The server won't start if there are not all attributes maintained in the repository key language.

The occurred message in the server log file will look like this:

```
The database 'HPM_MAIN' contains invalid structure feature entries. '3' entries have
been found which aren't maintained in the repository language German.
```

1.4.2.1 Solution

Create the entries for the missing language. The provided script is checking the item attributes and structure attributes. It will lookup of the missing data entries and copy an existing entry with the lowest language ID for the missing repository language.

Because the name of the created attribute has to be unique, the name will be '< name of the attribute to copy >< defined delimiter in the cmd file >< id of the attribute to copy >'. This name will be cut at the cmd defined length (by default DB 'Name' column length).



It is possible that this script creates invalid data if the attribute name of the the attribute to copy is as long as the defined max length in the cmd file and this name is already used by another attribute in the same context.

createMissingAttributes_mssql_pim7.rar

createMissingAttributes_oracle_pim7.rar

createMissingAttributes_mssql_pim8.rar

createMissingAttributes_oracle_pim8.rar

1.4.2.2 Consequences

The repository language **MUST NOT** be changed as soon as entity data such as items/products/variants or structures/structure groups have been created and exist in the database. In such a situation, the stability of the system can no longer be guaranteed since logical key fields most likely will contain null values.

1.4.3 MS-SQL Server

1.4.3.1 "The query processor could not start the necessary thread resources for parallel query execution"

During high-end benchmark tests the following error message has been thrown by the SQL Server " The query processor could not start the necessary thread resources for parallel query execution ". After some research we came to the conclusion that the only currently known solution to this problem is to reduce the load on the database server. Several articles from Microsoft or Microsoft close resources explained this to be the only way to avoid this problem. See also this article from Microsoft: <http://msdn.microsoft.com/en-us/library/aa337448.aspx>

So, how to reduce the load on the database server? First approach is to reduce the number of concurrent connections using the [DB Connection pool](#) (see page 56) settings as described in the Tuning Advisory. Additionally to that please always inform the Informatica Software Support. It will be very helpful if you can describe which action lead to the error, steps to reproduce would be great too since we're constantly investigating this issue and try to find a better solution.

In case this problem occurs often on your system you might want to observe the processor queue, this can give us hints on the processor utilisation at the time the error occurred.

1.4.3.2 "Transaction (Process ID XXX) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction"

In case you encounter database deadlocks you can use SQL Server Profiler to find the cause for the particular deadlock.

You need to run a server-side trace to capture deadlock information, you can use the script available here: <http://www.sqlservercentral.com/scripts/deadlock/66808/>

Once the deadlock has occurred, you can read the trace file with the script available here: <http://www.sqlservercentral.com/scripts/deadlock/66809/>

The content of the trace file will show which transactions resp. which SQL statements are responsible for the deadlock so further actions can be taken.

1.4.4 Oracle

1.4.4.1 Import/Export of schemas

To obtain some prepared notes on how to import export schemas from/to an Oracle instance, please raise a request with Informatica.

1.4.4.2 Connection Issues

If your project is having sporadic connection issues with Oracle database maybe these topics are helpful:

1. Maximum amount of processes. Because there were more than one application on the Oracle database the number of 300 was too small. We had good experiences with 1000 (minimum 300 x count of applications)
2. open_cursors, also too small values. We had good experiences with 1000 (minimum 300 x count of applications)
3. Log-Files are full. Mostly the Listener log file.



Switch off listener logging

Carry out following steps to configure the listener logging:

- a. Open [listener.ora](#) in `..\product\11.2.0\dbhome_1\NETWORK\ADMIN`
- b. Add **LOGGING_<Listernername> = OFF**
- c. Replace `<Listernname>` through your `listernname` e.g. 11GR2
- d. Save the changes
- e. Restart db and listener service.

Possible Values:

[ON| OFF]

4. tempdb full. Index rebuild after installation of standard classification systems!
5. Audit-Trail: waste logs also and makes them full. Towards description of Oracle the parameter „audit_trail“ is NOT deactivated per default! We deactivated this parameter on all of our instances.



Attention: when applying changes via enterprise manager console please ensure that the changes are applied in spfile, too.

1.4.4.3 Oracle JDBC driver OutOfMemoryError / huge memory consumption

Oracle JDBC driver may course OutOfMemoryError. The reason is that Oracle driver tries to allocate huge buffer for the incoming data. Sizes of these buffers depend on a single result set row size and 'prefetchRowSize' parameter of the prepared statement. By default this parameter is set to 10, but to increase list model performance it has been set to 10000 for all fragment SPs calls.

There are following different ways to avoid OutOfMemoryError in the oracle jdbc driver:

- Use jdbc driver parameter to limit buffer size in the UDA configuration file:

```
<property name="connectionProperties">oracle.jdbc.maxCachedBufferSize=<sizeInBytes>;</property>
```

- Set 'rowPrefetchSize' in the com.heiler.ppm.fragment.server.fragment extension point to override the global value for a certain fragment
- Set 'defaultRowPrefetchSize' in the com.heiler.ppm.fragment.server/preferences.ini or in the plugin_customization.ini

```
# Global value to control number of rows that will be fetch from the database in one
round-trip
# using fragment stored procedure. It helps jdbc driver to find trade-off between
memory consumption
# and db fetch performance. Default value set by many jdbc drivers is 10, but to
improve list model data access
# we need to set this value to a much higher value.
#
# Some JDBC driver implementations use this value to estimate incoming raw data
buffer size as
# defaultRowPrefetchSize*rowSize. Such buffers may have very large size (several
hunderd megabyte).
# Set this parameter to a low value if application server has small amount
# of memory. You can also experiment with this value if garbage collection pessure is
too high.
#
# This global value can be overridden per fragment using optional 'rowPrefetchSize'
attribute of the
# com.heiler.ppm.article.core.server.Fragment extension point. It is recommended to
use this attribute if
# the average number of rows returned by the fragment can be estimated in development
time.
# defaultRowPrefetchSize = 10000
```

See Oracle JDBC MemoryManagement for more info

1.4.4.4 Oracle JDBC driver OutOfMemoryError (Cache)

The listModel (Product 360 - Server Table) access with NClobs may result in an OutOfMemoryError.

The reason of this dump is oracle driver cache for CLOB fields. JDBC statements hold references to huge byte arrays. This strategy usually reduces number of DB round-trips and improves performance but it takes a lot of memory.

Try setting **oracle.jdbc.FreeMemoryOnEnterImplicitCache** connection property to true. This will instruct Oracle driver to do cache clean-up more frequently.

see oracle driver doc for more detail: http://download.oracle.com/docs/cd/E14072_01/appdev.112/e13995/oracle/jdbc/pool/OracleDataSource.html

1.4.4.5 Oracle ReportStoreTemp performance

Product 360 - Server reporting framework uses ReportStore and ReportStoreTemp tables to save report ids. Under heavy load report save rate can be very high and ReportStoreTemp tables may grow up to dozens of million records.

If possible we recommend putting these tables on a high performance storage.

1.4.4.6 Automatic expiration of passwords for Oracle DB users

When setting up the Oracle DB and the users needed for Product 360 it is important to not set the password of each of them to automatically expire. Once the expiration date is met the Product 360 Server and components will not be able to start up anymore.

1.5 Diagnosing and tuning poorly performing Oracle queries

This is a big topic, but we'll focus in on what a bad query plan looks like, what a good query plan looks like, and one example of how to improve a poor performing query. There is no generic way to tune all poorly performing queries, but hopefully this page can educate you on the tools you need to identify and diagnose a poor performing query and an example of how this particular query was tuned. The first part of this how-to is PIM specific, but the important parts are only specific to Oracle.

The first step is to find the poor performing query. In our example, a PIM Desktop user issued a query and noticed poor performance:

☐ Filter by assortment

Find field: LT Product

Comparison operator:

Comparison value:

Find field: Group ID (LT)

Comparison operator:

Comparison value:

☒ Match case

Find field: Attribute value of Heel Height

Comparison operator:

Comparison value:

☒ Match case

☐ Display result in new view

The first step is to go through the logs to try to find the query. As PIM uses hibernate, the following categories (`org.hibernate.SQL` and `org.hibernate.type.descriptor.sql.BasicBinder`) had to be updated to `TRACE` in the `log4j.xml` file to enable logging of SQLs as well as bind parameters:

```
<!-- Use "TRACE" to see the debug output from UDA and activate tracing in stored
procedures.
    Check also server.properties property "db.default.debug.show_sql" -->
<category name="UDA" additivity="false">
  <priority value="TRACE"/>
  <appender-ref ref="UdaAppender"/>
</category>

<!-- Use "TRACE" to see the debug output from Hibernate
    org.hibernate.cache           Log all second-level cache activity
    org.hibernate.hql.ast.AST     Log HQL and SQL ASTs during query parsing
    org.hibernate.jdbc            Log all JDBC resource acquisition
    org.hibernate.pretty          Log the state of all entities (max 20 entities)
associated with the session at flush time
    org.hibernate.SQL             Log all SQL DML statements as they are executed
(include org.hibernate.type to see binds and values)
    org.hibernate.secure          Log all JAAS authorization requests
    org.hibernate.tool.hbm2ddl    Log all SQL DDL statements as they are executed
```

```

    org.hibernate.transaction    Log transaction related activity
    org.hibernate.type           Log all JDBC parameters
    org.hibernate                Log everything -->
<category name="org.hibernate.SQL" additivity="false">
  <level value="TRACE" />
  <appender-ref ref="UdaAppender"/>
</category>
<category name="org.hibernate.type.descriptor.sql.BasicBinder" additivity="false">
  <level value="TRACE" />
  <appender-ref ref="UdaAppender"/>
</category>

```

Changes in this file do not require a server restart, and after the above query is executed again, you will need to find the query in the logs.

Here's the query after we've replaced the necessary bind parameters:

```

set linesize 1000

explain plan for
select articlerev0_."ID", articlerev0_."Identifier"
  from
    "ArticleRevision" partition(P_PRODUCT) articlerev0_
 left outer join
    "ArticleDetail" partition(P_PRODUCT) articledet1_
      on articlerev0_."ID"=articledet1_."ArticleRevisionID"
      and articledet1_."DeletionTimestamp" = TO_TIMESTAMP('9999-12-31','yyy
y-mm-dd')
 left outer join
    "ArticleLang" partition(P_PRODUCT) articlelan2_
      on articlerev0_."ID"=articlelan2_."ArticleRevisionID"
      and (
        articlelan2_."LanguageID"=9
        and articlelan2_."Res_LK_Text100_01"='LT'
        and articlelan2_."Res_LK_Int_01"=0
        and articlelan2_."ChannelID"=1
        and articlelan2_."EntityID"=21007
      )
      and articlelan2_."DeletionTimestamp" = TO_TIMESTAMP('9999-12-31','yyy
y-mm-dd')
 left outer join
    "ArticleAttribute" partition(P_PRODUCT) articleatt3_
      on articlerev0_."ID"=articleatt3_."ArticleRevisionID"
      and (
        articleatt3_."NameInKeyLanguage"='Heel Type'
      )
      and articleatt3_."DeletionTimestamp" = TO_TIMESTAMP('9999-12-31','yyy
y-mm-dd')
 left outer join
    "ArticleAttributeValue" partition(P_PRODUCT) articleatt4_
      on articleatt3_."ID"=articleatt4_."ArticleAttributeID"
      and (

```

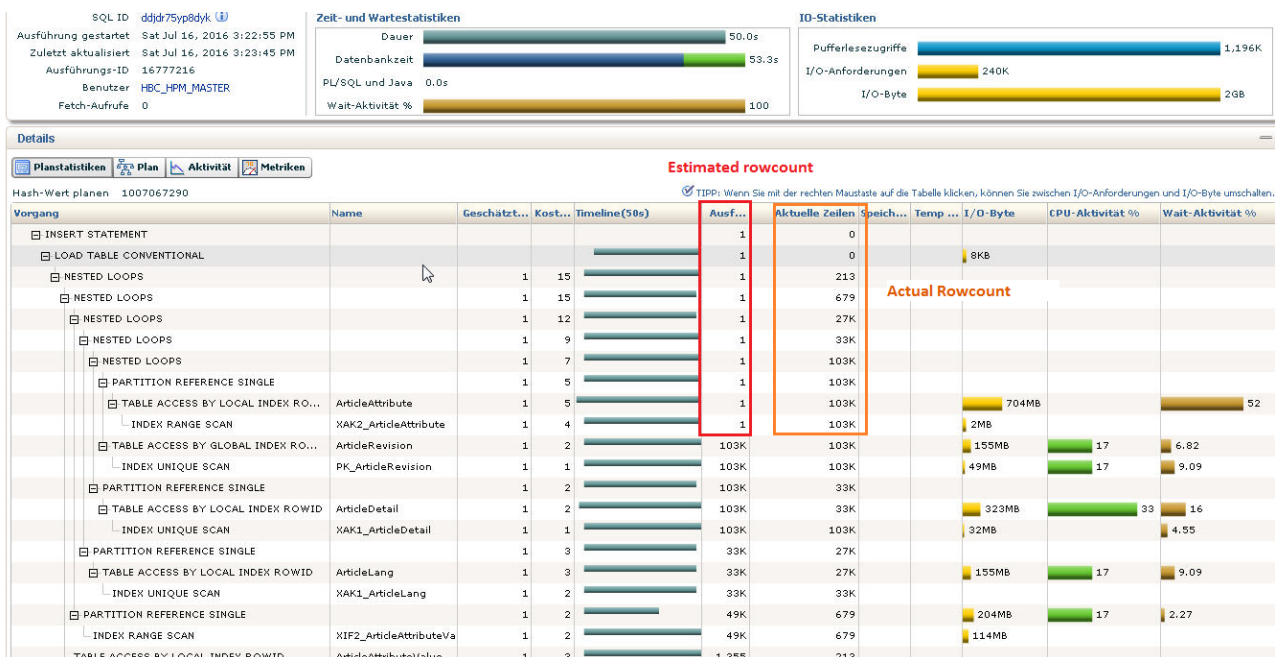


```

        articleatt4."Identifier"='ALL'
        and articleatt4."LanguageID"=9
    )
    and articleatt4."DeletionTimestamp" = TO_TIMESTAMP('9999-12-31','yyy
y-mm-dd')
    where
        articlerev0."RevisionID" = 1
        and articlerev0."DeletionTimestamp" = TO_TIMESTAMP('9999-12-31','yyyy-
mm-dd')
        and articledet1."Res_Bit_02"=1
        and articlelan2."Res_Text250_05"='41'
        and (
            articleatt4."Value" is not null
        )
        and articlerev0."EntityID"=1100;

```

Explaining the plan query for this resulted in the following. Notice the difference between the estimated row count and the actual row count? This results in Oracle choosing an incorrect plan with an incorrect join order and many nested loops.



Also notice that an INDEX RANGE SCAN was performed on index XAK2_ArticleAttribute in table ArticleAttribute, which resulted in an incorrect estimated row count.

To investigate further, a 10053 Optimizer trace is needed. In the same session from where you issued the query and explain plan, do the following. Pay attention to the commented lines below. You'll need to turn on tracing, execute the query, then turn off tracing. Then you'll need to view the trace files, preferably with a utility:

```
-- activate generating trace files
```

```

ALTER SESSION SET EVENTS='10053 trace name context forever, level 1';
-- deactive generating trace files
ALTER SESSION SET EVENTS '10053 trace name context off';
-- get filenames for trace session
select u_dump.value || '/' || instance.value || '_ora_' || v$process.spid
|| nvl2(v$process.traceid, '_' || v$process.traceid, null ) || '.trc'"Trace File"
from V$PARAMETER u_dump
cross join V$PARAMETER instance
cross join V$PROCESS
join V$SESSION on v$process.addr = V$SESSION.paddr
where u_dump.name = 'user_dump_dest'
and instance.name = 'instance_name'
and V$SESSION.audsid=sys_context('userenv','sessionid');

```

Navigate to the Explain Plan section of the trace file to confirm you are looking at the proper query in the trace file:

```

----- Explain Plan Dump -----
----- Plan Table -----

=====
Plan Table
=====
-----
+-----+-----+
| Id | Operation | Name | Rows |
|-----+-----+
| 0 | SELECT STATEMENT | | |
| 10 | | | |
| 1 | NESTED LOOPS | | 1 |
116 | 10 | 00:00:01 | | |
| 2 | NESTED LOOPS | | 1 |
116 | 10 | 00:00:01 | | |
| 3 | NESTED LOOPS | | 1 |
84 | 7 | 00:00:01 | | |
| 4 | PARTITION REFERENCE SINGLE | | 1 |
55 | 5 | 00:00:01 | KEY(AP) | KEY(AP) |
| 5 | TABLE ACCESS BY LOCAL INDEX ROWID | ArticleAttribute | 1 |
55 | 5 | 00:00:01 | 2 | 2 |
| 6 | INDEX RANGE SCAN | XAK2_ArticleAttribute | 1 |
| 4 | 00:00:01 | 2 | 2 |
| 7 | TABLE ACCESS BY GLOBAL INDEX ROWID | ArticleRevision | 1 |
29 | 2 | 00:00:01 | 2 | 2 |
| 8 | INDEX UNIQUE SCAN | PK_ArticleRevision | 1 |
| 1 | 00:00:01 | | |
| 9 | PARTITION REFERENCE SINGLE | | 1 |
| 2 | 00:00:01 | KEY(AP) | KEY(AP) |
| 10 | INDEX UNIQUE SCAN | XAK1_ArticleAttributeValue | 1 |
| 2 | 00:00:01 | KEY(AP) | KEY(AP) |

```

```

| 11 | TABLE ACCESS BY LOCAL INDEX ROWID | ArticleAttributeValue | 1 |
32 | 3 | 00:00:01 | 1 | 1 |
-----
+-----+

```

In my trace file the explain plan dump for the query starts at line 6613 .

Using a trace file viewer is recommended - general steps to find the issue are:

1. Find the proper 'optimizing query block' (OPTIMIZING QB) section for your sql (If using v10053.exe look for the OPTIMIZING QB section on the left hand side).
2. Look for the Access path analysis for the table containing the wrong statistic (ArticleAttribute in our case).
3. Look for the column containing the index giving the wrong estimated row count. In our case, the XAK2_ArticleAttribute index is on the NameInKeyLanguage column.

At this point you'll want to familiarize yourself with how Oracle uses histograms to manage statistics.

The screenshot shows the 10053 trace file viewer interface. On the left, the 'OPTIMIZING QB' section is highlighted with a red box and labeled '1'. The main window displays the 'Access path analysis for ArticleAttribute' section, which is highlighted with a red box and labeled '2'. This section shows the 'SINGLE TABLE ACCESS PATH' for 'ArticleAttribute' and the 'Column (#53)' statistics. The 'Column (#53)' statistics show a histogram with 255 buckets and a prorate density of 0.000000. The 'Using prorate density' message is highlighted in red and labeled '3'.

If you look closely at section 3, you'll notice that the NDV (Number of Distinct Values) is 383, but the #Bkts (Buckets) is 255 from the HtBal (Height Balanced) histogram. The NameInKeyLanguage value of 'Heel Height' was unfortunately not in one of the buckets, which causes the "Using prorate density: 0.000000 of col #54 as selectivity of out-of-range/non-existent value pred" message you see. This causes the optimizer to incorrectly guess that this value doesn't exist in this table and that joining to this table would also return no rows for the query (prorate density of 0).

To fix this, we need to delete the histogram. Here's a script which deletes all histograms for all varchars with > 254 distinct values, as the number of buckets isn't sufficient to cover all values so a height balanced histogram isn't desired for this scenario (in general, height balanced histograms are almost useless for varchar2 columns).

Script which deletes all height balanced histograms for nvarchar2 columns with more than 254 distinct values

```
-- script which deletes all height balanced histograms for nvarchar2 columns with
more than 254 distinct values
BEGIN
  FOR COL_ITEM IN (
    SELECT stats.OWNER, stats.TABLE_NAME, stats.COLUMN_NAME
    FROM DBA_TAB_COL_STATISTICS stats
    INNER JOIN DBA_TAB_COLUMNS cols
    ON stats.TABLE_NAME = cols.TABLE_NAME
    AND stats.COLUMN_NAME = cols.COLUMN_NAME
    WHERE cols.DATA_TYPE = 'NVARCHAR2'
    AND stats.HISTOGRAM <> 'NONE'
    AND stats.NUM_DISTINCT > 254
    AND stats.OWNER = cols.OWNER
    AND stats.OWNER IN ('HBC_HPM_MAIN', 'HBC_HPM_MASTER', 'HBC_HPM_SUPP
LIER')
    ORDER BY stats.OWNER, stats.TABLE_NAME, stats.COLUMN_NAME
  )
  LOOP
    DBMS_STATS.DELETE_COLUMN_STATS (OWNNAME=>COL_ITEM.OWNER, TABNAME=>' ' ||
COL_ITEM.TABLE_NAME || ' ', COLNAME=>' ' || COL_ITEM.COLUMN_NAME || ' ',
COL_STAT_TYPE=>'HISTOGRAM');
  END LOOP;
END;
/
```

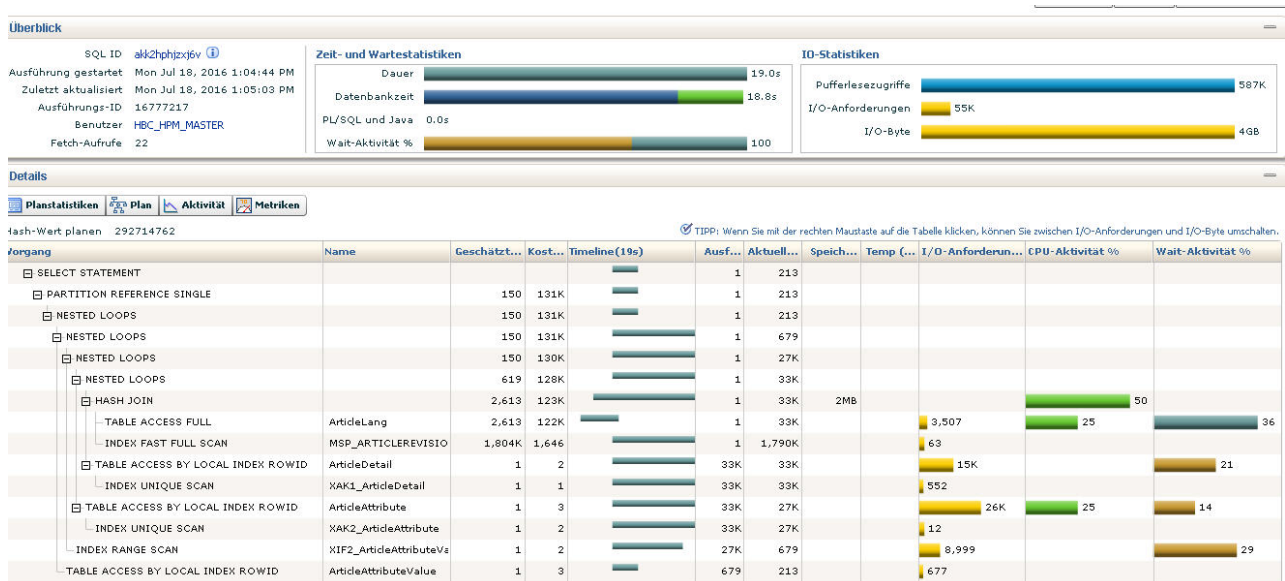
Finally, to prevent Oracle's automatic statistics job from regenerating the histogram you need to call the `set_table_prefs` function.

Pass size 1 as the `METHOD_OPT` to prevent histogram creation:

```
--This does not work as column and table name are lowercase and must be passed with
double quote.
--TODO is to fix this call to work with mixed case table and column names
BEGIN
  DBMS_STATS.set_table_prefs('HBC_HPM_MASTER', '"ArticleAttribute"', 'METHOD_OPT',
'for all columns size skewonly for columns size 1 "NameInKeyLanguage"');
END;
/
```

After fixing this and a similar issue in the `Res_Text250_05` column for table `ArticleLang`, we get the following plan. While the actual and estimated row counts are off, the join order is now more efficient (Good: `ArticleLang`, `ArticleDetail`, `ArticleAttribute`, `ArticleAttributeValue` vs. Bad: `ArticleAttribute`, `ArticleDetail`, `ArticleLang`, `ArticleAttributeValue` previously). For outer joins in general, the most limiting tables should be joined first, which for PIM means

`ArticleAttribute` and `ArticleAttributeValue` should usually be joined last as they are larger than the `Article` table by some factor.



1.5.1 Fixing frequency histogram issues

The second issue look for the `Res_Text250_05` column for table `ArticleLang`:

```
Column (#13):
NewDensity:0.000053, OldDensity:0.000000 BktCnt:9392, PopBktCnt:9375, PopValCnt:114,
NDV:208
Column (#13):
NewDensity:0.000122, OldDensity:0.000000 BktCnt:4113, PopBktCnt:4091, PopValCnt:103,
NDV:208
Column (#13): Res_Text250_05( Part#: 1
AvgLen: 4 NDV: 208 Nulls: 10118996 Density: 0.000122
Histogram: Freq #Bkts: 125 UncompBkts: 4113 EndPtVals: 125
Column (#13): Res_Text250_05(
AvgLen: 4 NDV: 208 Nulls: 10118996 Density: 0.000122
Histogram: Freq #Bkts: 125 UncompBkts: 4113 EndPtVals: 125
...
...
Using density: 0.000122 of col #13 as selectivity of unpopular value pred
```

Issue: The value selected did not fall into any of the histogram buckets.

Solution: Add more buckets such that all values in the column can be assigned a bucket by forcing the optimizer to analyze 100 percent of the rows. As the number of distinct values in this column is less than 254 (Oracle max # of buckets) this is a suitable solution:

```
-- script which adds more buckets for frequency histograms
```

```

DECLARE
    BUCKETS_BUFFER NUMBER := 0;
BEGIN
    FOR COL_ITEM IN (
        SELECT stats.OWNER, stats.TABLE_NAME,
        LISTAGG('FOR COLUMNS "' || stats.COLUMN_NAME || '" SIZE ' || CASE
            WHEN stats.NUM_DISTINCT + BUCKETS_BUFFER > 254 THEN 254 ELSE stats.NUM_DISTINCT +
            BUCKETS_BUFFER END, ', ') WITHIN GROUP (ORDER BY stats.COLUMN_NAME) AS METHOD_OPT
        FROM DBA_TAB_COL_STATISTICS stats
        INNER JOIN DBA_TAB_COLUMNS cols
        ON stats.TABLE_NAME = cols.TABLE_NAME
        AND stats.COLUMN_NAME = cols.COLUMN_NAME
        WHERE cols.DATA_TYPE = 'NVARCHAR2'
        AND stats.HISTOGRAM = 'FREQUENCY'
        AND stats.NUM_DISTINCT > stats.NUM_BUCKETS
        AND stats.OWNER = cols.OWNER
        AND stats.OWNER IN ('HBC_HPM_MAIN', 'HBC_HPM_MASTER', 'HBC_HPM_SUPP
    LIER')

        GROUP BY stats.OWNER, stats.TABLE_NAME
        ORDER BY stats.OWNER, stats.TABLE_NAME
    )
    LOOP
        DBMS_STATS.GATHER_TABLE_STATUS
        (
            OWNNAME => COL_ITEM.OWNER,
            TABNAME => '"' || COL_ITEM.TABLE_NAME || '"',
            ESTIMATE_PERCENT => 100,
            METHOD_OPT => COL_ITEM.METHOD_OPT,
            CASCADE => FALSE
        );
    END LOOP;
END;
/

```

1.6 Oracle Import/Export for MAIN, MASTER and SUPPLIER Schemas

A "how to" guideline for exporting and importing Product Manager Schemas from and to an Oracle 11G.

Informatica Software Development or Support might ask you to provide the data of a certain installation in order to reproduce bugs or performance issues. This guide shall help support, partners and consulting to provide this.

- [Preparation](#) (see page 39)
 - [Directory Object](#) (see page 39)
- [Export](#) (see page 39)
 - [User Roles](#) (see page 39)
 - [Execute export datapump](#) (see page 40)
- [Import](#) (see page 40)
 - [User Roles](#) (see page 40)
 - [Tablespaces](#) (see page 40)
 - [Execute Import Datapump](#) (see page 41)



The terms backup/restore which are usually used in the SQL Server world map to export/import in Oracle. The actual backup/restore functionality of Oracle is bound to a specific oracle instance (more or less) and is only meant for error/data loss recovery, but not to transfer the data to a different database instance.

1.6.1 Preparation

The whole export/import process can not be performed by the sys user at all. Therefore we suggest to create a new user for administrative purposes. In the following sections we will refer to this user as `hlradmin`. The effective permissions a user needs to have differ between the export and import process, thus they are described in the corresponding sections.

Create the hlradmin user

```
1 CREATE USER "HLRADMIN" IDENTIFIED BY "heiler" PROFILE "DEFAULT" ACCOUNT UNLOCK;
```

1.6.1.1 Directory Object

The directory object in oracle maps to a physical directory path. This directory is used to store the dumpfile(s). The directory object is needed for the export as well as the import.

Create the directory object

```
1 CREATE DIRECTORY "HLR_DIR" AS 'c:\heiler\oracle\impexp'
```

1.6.2 Export

1.6.2.1 User Roles

The `hlradmin` user needs the following privileges to export data

- resource
- connect
- select permissions on all objects for the MAIN, MASTER and SUPPLIER users

Grant needed privileges to hlradmin user

```
1 GRANT CONNECT TO "HLRADMIN";
2 GRANT RESOURCE TO "HLRADMIN";
```

```
3 GRANT EXP_FULL_DATABASE TO "HLRADMIN";
```

1.6.2.2 Execute export datapump

1.6.3 Import

1.6.3.1 User Roles

The hlradmin user needs the following privileges to perform the import:

- imp_full_database
- create user
- create any table
- create any index
- create any sequence
- create any procedure
- create tablespace

Grant needed privileges to hlradmin user

```
1 GRANT IMP_FULL_DATABASE TO "HLRADMIN";
2 GRANT CREATE USER TO "HLRADMIN";
3 GRANT CREATE ANY TABLE TO "HLRADMIN";
4 GRANT CREATE TABLE WITH ADMIN OPTION TO "HLRADMIN";
5 GRANT CREATE ANY INDEX TO "HLRADMIN";
6 GRANT CREATE INDEX WITH ADMIN OPTION TO "HLRADMIN";
7 GRANT CREATE ANY SEQUENCE TO "HLRADMIN";
8 GRANT CREATE SEQUENCE WITH ADMIN OPTION TO "HLRADMIN";
9 GRANT CREATE ANY PROCEDURE TO "HLRADMIN";
10 GRANT CREATE PROCEDURE WITH ADMIN OPTION TO "HLRADMIN";
11 GRANT CREATE TABLESPACE TO "HLRADMIN";
```

1.6.3.2 Tablespaces

To import a dump file which contains the tablespaces of the MAIN, MASTER and SUPPLIER schemas, you need to have corresponding tablespaces in the target database. You can execute the following pl/sql snippet for each needed schema in order to create the tablespaces.

Create empty tablespaces

```
1 -- Temporary tablespace
2 CREATE TEMPORARY TABLESPACE "<SCHEMA_NAME>_TEMP"
3     TEMPFILE 'c:/heiler/oracle/impexp/<SCHEMA_NAME>_Temp.dbf'
4     SIZE 1024M REUSE
```



```

5      AUTOEXTEND ON NEXT 128M
6      MAXSIZE UNLIMITED
7      EXTENT MANAGEMENT LOCAL
8      UNIFORM SIZE 1M;
9      -- Data tablespace
10     CREATE TABLESPACE "<SCHEMA_NAME>_DATA"
11         DATAFILE 'c:/heiler/oracle/impexp/<SCHEMA_NAME>_Data.dbf'
12         SIZE 1024M REUSE
13         AUTOEXTEND ON NEXT 128M
14         MAXSIZE UNLIMITED
15         EXTENT MANAGEMENT LOCAL
16         SEGMENT SPACE MANAGEMENT AUTO;
17     -- Separate index tablespace
18     CREATE TABLESPACE "<SCHEMA_NAME>_INDEX"
19         DATAFILE 'c:/heiler/oracle/impexp/<SCHEMA_NAME>_Index.dbf'
20         SIZE 1024M REUSE
21         AUTOEXTEND ON NEXT 128M
22         MAXSIZE UNLIMITED
23         EXTENT MANAGEMENT LOCAL
24         SEGMENT SPACE MANAGEMENT AUTO;

```

1.6.3.3 Execute Import Datapump

Parameter file

```

1      # the directory object
2      DIRECTORY=HLR_DIR
3      # the name of the ora dump file, located in the physical file directory
   mapped to DIRECTORY
4      DUMPFILE=MYEXPORT01.DMP
5      # the schemas to be imported
6      # attention: you have to import all PIM - Server schemas (there're
   dependencies between the schema objects)
7      SCHEMAS=CUST_MAIN,CUST_MASTER,CUST_SUPPLIER
8      # specify the schema name mappings, necessary if schema names to be
   imported are different from local schema names
9      # schemas are created automatically if they don't exist
10     # pattern:
11     # schema name in dump file:local schema name
   REMAP_SCHEMA=CUST_MAIN:HPM_MAIN,CUST_MASTER:HPM_MASTER,CUST_SUPPLIER:HPM_S
   UPPLIER
12     # specify the tablespace name mappings, necessary if tablespace names to
   be imported are different from local tablespace names
13     # tablespace are NOT created automatically, you have to create them
   before
14     # pattern:
15     # tablespace name in dump file:local tablespace name (must exist)
   REMAP_TABLESPACE=
16     CUST_MAIN_INDEX:HPM_MAIN_INDEX,CUST_MAIN_DATA:HPM_MAIN_DATA,CUST_MAIN_TEMP
   :HPM_MAIN_TEMP,CUST_MASTER_INDEX:HPM_MASTER_INDEX,CUST_MASTER_DATA:HPM_MAS

```

```

TER_DATA,CUST_MASTER_TEMP:HPM_MASTER_TEMP,
CUST_SUPPLIER_INDEX:HPM_SUPPLIER_INDEX,CUST_SUPPLIER_DATA:HPM_SUPPLIER_DATA,
CUST_SUPPLIER_TEMP:HPM_SUPPLIER_TEMP
17 # use this option to only create the sql file
18 # no import will be executed
19 # SQLFILE=SQL_IMPORT.SQL

```

execute import cmd

```
1 impdp USERID=HLRADMIN/heiler parfile="<path to parameter file>" | pause
```



If you get errors regarding insufficient access rights of user HLRADMIN, try to execute with SYSTEM-User:

```
impdp USERID=SYSTEM/heiler parfile="<path to parameter file>" | pause
```



While importing, you might get several Oracle errors regarding the users .._HPM_BULKLOAD and .._HPM_REPSTORE:

ORA-39083: Objekttyp OBJECT_GRANT konnte nicht erstellt werden, Fehler:

ORA-01917: Benutzer oder Funktion KRAMP_HPM_REPSTORE ist nicht vorhanden

Fehlerhafte SQL ist:

```
GRANT FLASHBACK ON "KRAMP_HPM_MAIN"."User" TO "KRAMP_HPM_REPSTORE"
```

ORA-39083: Objekttyp OBJECT_GRANT konnte nicht erstellt werden, Fehler:

ORA-01917: Benutzer oder Funktion KRAMP_HPM_BULKLOAD ist nicht vorhanden

Fehlerhafte SQL ist:

```
GRANT ALTER ON "KRAMP_HPM_MAIN"."UnitSystemUnitMapLang" TO
"KRAMP_HPM_BULKLOAD"
```

They do not have any impact on the import result, so you can ignore them.

2 Server Operation

Combines maintenance, tuning and troubleshooting for the Product 360 application server.

- [Startup & Shutdown](#) (see page 43)
 - [Status Cache](#) (see page 43)
- [Maintenance Tasks](#) (see page 43)
- [Monitoring](#) (see page 44)

- [Tuning](#) (see page 44)
 - [Memory Settings](#) (see page 44)
 - [Garbage Collection](#) (see page 44)
 - [GC activity log](#) (see page 45)
 - [Analyse GC activity log & optimize](#) (see page 45)
 - [Communication Framework](#) (see page 45)
 - [Mass-Data Parallel Processing](#) (see page 45)
 - [Media Asset Parallel Management](#) (see page 46)
 - [CreateZipManagement](#) (see page 46)
 - [MediaAssetProvider special MBean](#) (see page 47)
 - [MediaAssetProviderParallelManager](#) (see page 47)
 - [HeilerClassic special MBeans](#) (see page 48)
 - [CreateThumbnailManager](#) (see page 48)
 - [HeilerClassicParallelManager](#) (see page 48)
 - [Database Connection Pool](#) (see page 49)
 - [Logging](#) (see page 51)
 - [Import](#) (see page 51)
 - [Preferences regarding CPU usage](#) (see page 51)
 - [Preferences regarding memory usage](#) (see page 53)
- [Trouble Shooting](#) (see page 53)
 - [Anti-Virus Software affecting performance](#) (see page 53)
 - [Media Asset Provider \(Classic Provider\)](#) (see page 53)
 - [No preview are visible](#) (see page 53)
 - [Obtain a Memory Dump \(Heap Dump\) from the Java Virtual Machine](#) (see page 54)
 - [Enhanced Logging for Assortment Content Calculation](#) (see page 54)
 - [Server and client in different time zones](#) (see page 55)
 - [Cluster](#) (see page 55)
 - [Usage of strong cryptographic algorithms to encrypt/decrypt secure information](#) (see page 55)

2.1 Startup & Shutdown

TODO MSpiegel general

2.1.1 Status Cache

TODO MSpiegel add description

[Status Cache Initialization](#) (see page 74)

2.2 Maintenance Tasks

- Backup Strategy
- Background job should be configured to be automatically deleted after a certain amount of time.
See [Server Job Maintenance](#) (see page 60) for details.

2.3 Monitoring

Monitoring tools which can communicate using the SNMP protocol can be used to monitor the Product 360 application servers. Please see [SNMP Monitoring](#) for configuration details on the SNMP functionality.

Additionally micrometer metrics are available. See page [Monitoring with Micrometer](#) (see page 79) for more information.

2.4 Tuning

Here is the list of settings and configuration options that may affect system performance. These settings depend on system workloads, hardware and environment.

2.4.1 Memory Settings

As expected, the memory settings are crucial for the application server of the PIM - Server. You can configure the available memory in the `<installation root>/_environment.conf` file. Please note that you can configure the maximum amount of memory (`MEM_MAX`).



The minimum amount of available heap memory will always be set to the same value as the `MEM_MAX` , since for an application server it makes no sense otherwise.

2.4.2 Garbage Collection

Garbage collection configuration is a very complicated thing to do. Informatica PIM Software engineers spend a lot of time on site with customers to find the right garbage collection algorithm and settings. All needed parameters have already been adjusted by us, the only configuration which is needed for every customer installation is the number of garbage collection threads which can be used. PIM - Server uses a parallel garbage collection algorithm, therefore you can decide how many threads the garbage collection can use. The `GC_THREADS` setting in the `<installation root>/_environment.conf` file should be set to about 75% of the available CPU cores. For example, in case you have 8 Cores (including hyper-threading cores), you would set this to about 6 threads.



In case you need to further adjust the garbage collection settings you can do so in the `<installation root>/service/wrapper.conf` file in the `GC_OPTIONS` section. Please note that the `console_debug.cmd` launch file will use different garbage collection algorithms since the debug mode is not able to handle the ones we use for production.

2.4.2.1 GC activity log

In order to analyze the GC events in detail you can use the GC activity log for each Product 360 Server located in the `<installation root>/logs` folder. GC activity logs are enabled per default for all Product 360 Servers.

Each Server start produces a new `gc_<timestamp>.log.*` file (whereas `<timestamp>` is the starting time of the Product 360 Server) in the server's log folder.

2.4.2.2 Analyse GC activity log & optimize

The GC activity log can be analyzed e.g. online via <http://gceasy.io/>. Common issues and optimization possibilities are:

- The "GC pause duration time" specifies the length of the time intervals, where the garbage collector runs exclusively and therefor pauses all other application threads. If the **"GC pause duration time" is above 10 seconds**, the Product 360 Server application runs into trouble, because there are continuously running heartbeats (e.g. for the Hazelcast Framework, which have a timeout of 10 seconds. This can result in operation timeouts in the log file (e.g. HazelcastOperationTimeout)
 - **For optimization: consider increasing heap memory** (MEM_MAX in `<installation root>/_environnement.conf` file)
- The "object creation rate" (for <http://gceasy.io/>: see table "Object stats", row "Avg creation rate") specifies the heap memory size, which is allocated per second by new created java objects. If there are problems with operation timeout exceptions in the log due to long "GC pause duration time" (see the explained issue above) AND the **"object creation rate" is also very high** (somewhere above 1 GB/s), then the garbage collector is maybe running too late and then is not able to free memory in less than 10 seconds. By default the garbage collector G1 begins to work, when 45% of the heap memory is in use. This is specified by the parameter "InitiatingHeapOccupancyPercent", which has a default value of 45.
 - **For optimization: consider decreasing the "InitiatingHeapOccupancyPercent"**. Therefor add `-XX:InitiatingHeapOccupancyPercent=30` (or even 20 or 10) to `GC OPTIONS` section in `<installation root>/service/wrapper.conf`

Find general information about the used Garbage collection algorithm G1 here: <https://www.oracle.com/technetwork/tutorials/tutorials-1876574.html>

2.4.3 Communication Framework

Number of worker threads in server network settings

2.4.4 Mass-Data Parallel Processing

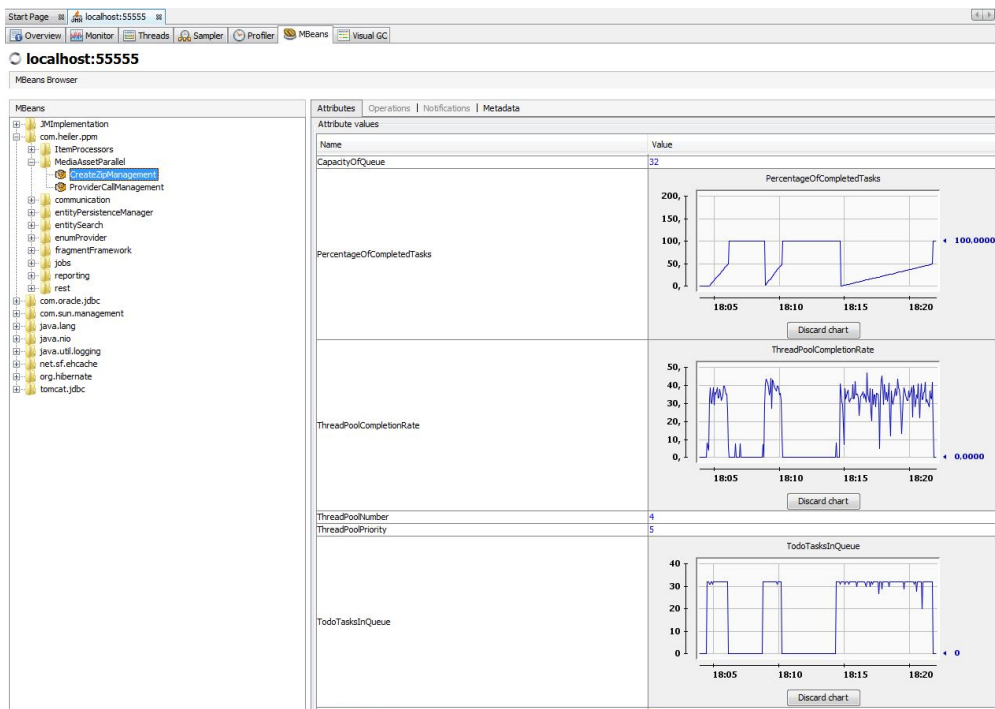
Item Processor Threadpool

2.4.5 Media Asset Parallel Management

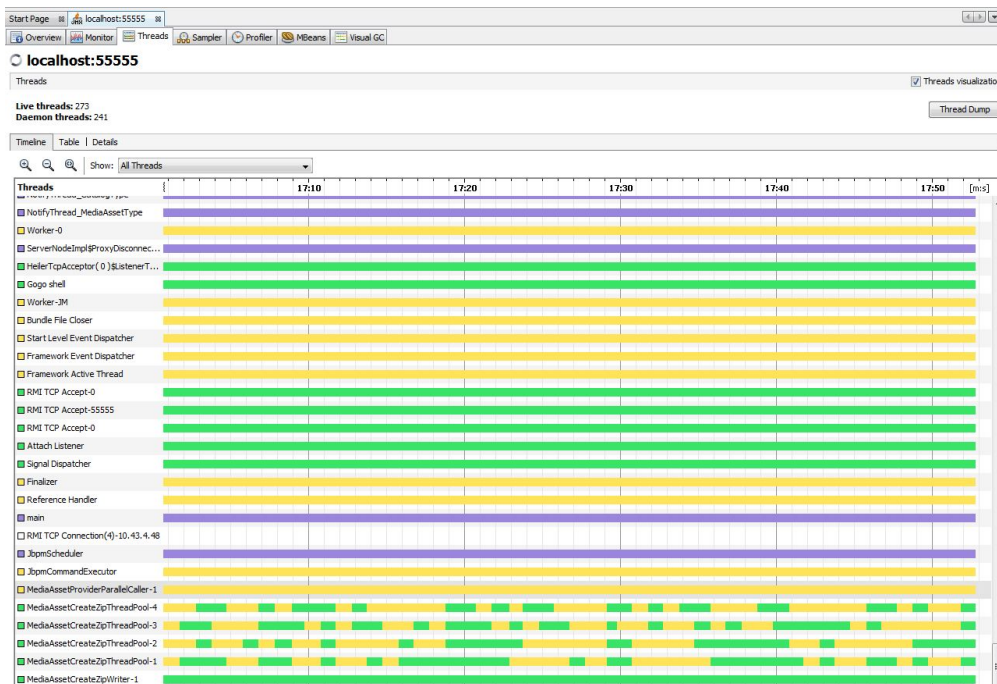
There are several registered MBeans under the category **com.heiler.ppm** → **MediaAssetParallel** with the JMX interface which are applied to observe and improve the performance in run time.

2.4.5.1 CreateZipManagement

It provides the management of the thread pool and other relevant parameters(e.g. the size of the BlockingQueue to avoid OutOfMemoryError) by zip creation phase.



The following screenshot indicates the corresponding threads in run time.



2.4.5.2 MediaAssetProvider special MBean

MediaAssetProviderParallelManager

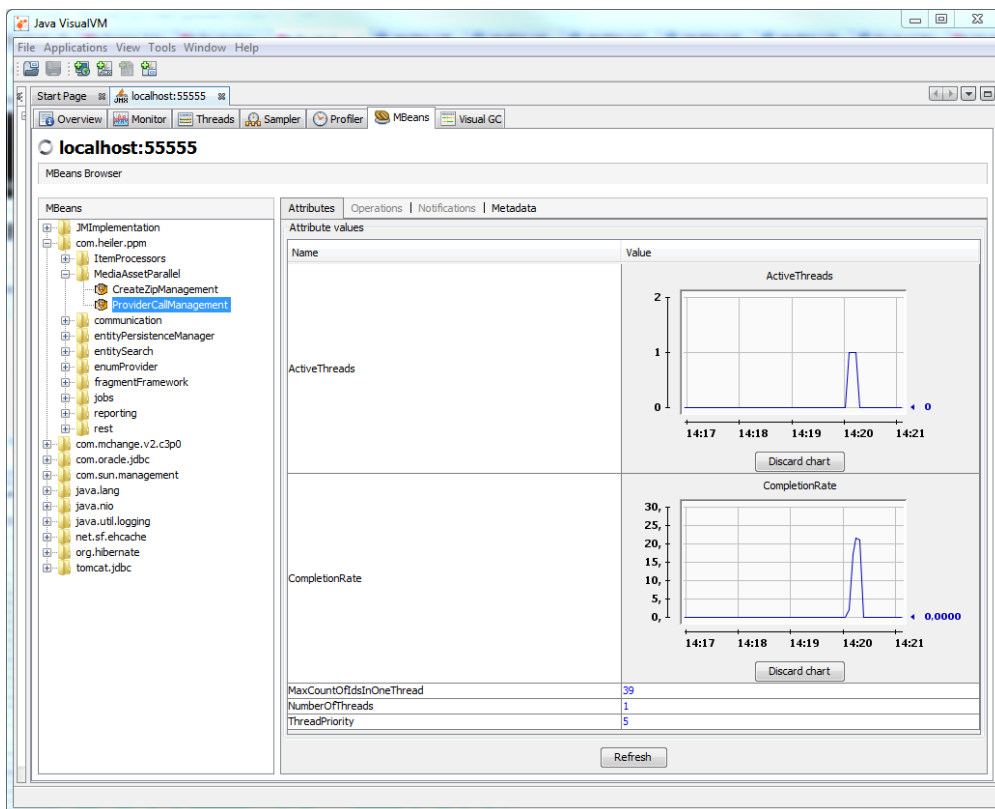


It was named as "ThreadPoolManagement", and changed as "ProviderCallManagement" and then as "MediaAssetProviderParallelManager" since 7.1.01.00.

It provides the management of the multi-threads call for the corresponding MediaAssetProvider methods.

A special thread pool is applied to improve the performance for some methods of the default IMediaAssetProvider, such methods should have any parameter composed of a long id list(e.g. getUNCpaths(String[] identifiers, ...)). Currently only the PIM - Media Manager 'HMM' MediaAssetProvider supports it. This thread pool settings can be observed and adjusted with the JMX interface during runtime. It is meanful for some scenarios like export with mass data.

The following screenshot indicates the thread pool utilization. The user can adjust the values of the attributes "MaxCountOfIdsInOneThread" and "NumberOfThreads" to improve the performance, if the "ActiveThreads" reaches always the maximum value and the "CompletionRate" is but too low.



2.4.5.3 HeilerClassic special MBeans



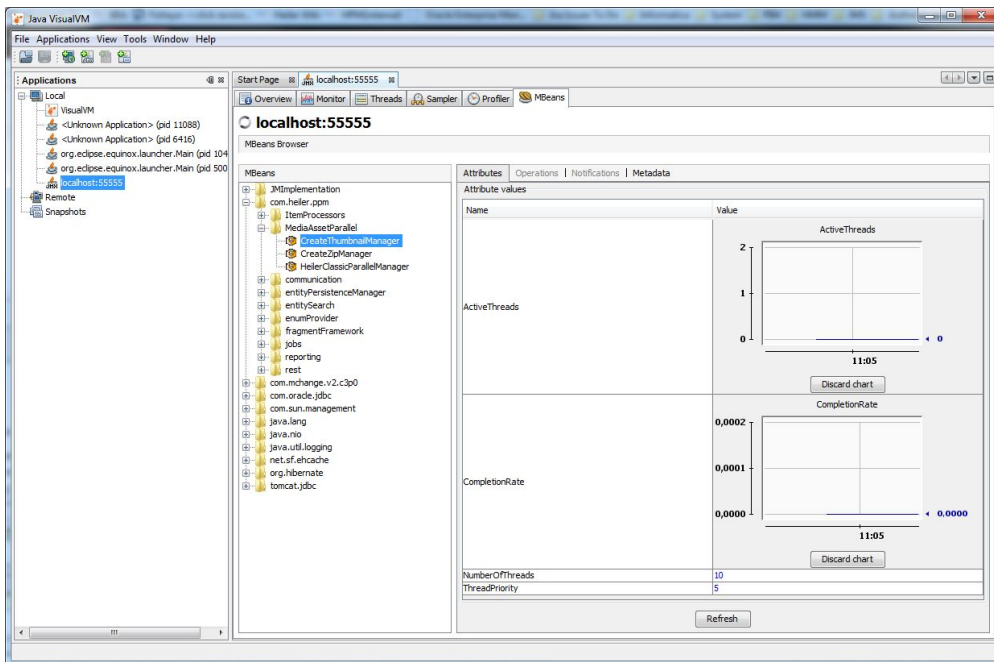
since 7.1.01.00 HeilerClassic registers two MBeans which can be also observed and adjusted with JMX interface.

CreateThumbnailManager

It provides the management of thread pool which schedules all graphicsmagick convert jobs to create thumbnails. The default NumberOfThread is 10.

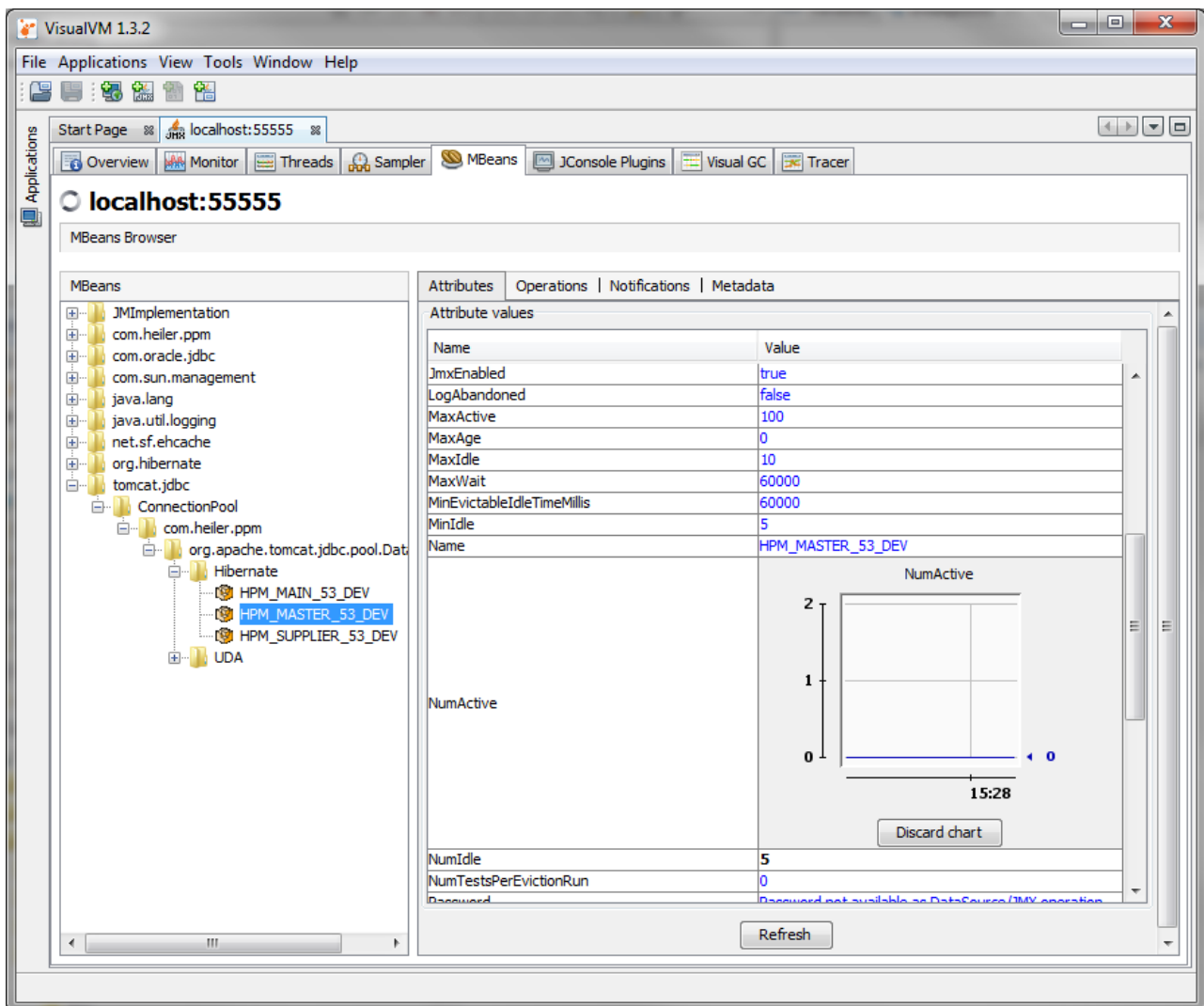
HeilerClassicParallelManager

It provides the management of thread pool which schedules tasks which can be performed parallelly for the corresponding MediaAssetProvider methods(e.g. GetSubCategoriesTask).



2.4.6 Database Connection Pool

The database connection pool settings can be adjusted in the corresponding configuration files. More detailed information on the configuration of the connection pool can be found here [DB Connection pool](#) (see page 56). During runtime the connection pool utilization can be observed with the JMX interface (see Screenshot). Most important here is the maximum used attribute which indicates if the maximum number of connections has already been used. You might need to increase the maximum connections in case the limit is reached often, or continuously.



This screenshot shows the `NumActive` attribute of the `ConnectionPool` for the `HPM_MASTER` connection pool. Currently there are zero connections in use. When this number reaches the `MaxActive` limit, the application load might be too high for the number of connections, adjusting the connection pool is necessary then. Additionally to `NumActive` and `MaxActive` you might want to check the `WaitCount`, which indicates how long a thread needs to wait until it acquires a connection from the pool. In case there are enough connections left in the pool, this number should be about zero.



The PIM - Server has two connection pool configurations which are of interest here, the Hibernate and the UDA one. The Hibernate pool is used for detail model persistence, therefore it's utilized in detail model modifications, import and merge. The UDA pool is used mostly for read-only mass-data retrieval like list model based functionality such as table views and export.

2.4.7 Logging

PIM - Server has a central configuration file for all logging activities in the server, as well as in the client. By default the logging trace levels are set to INFO, thus logging messages with severity INFO, WARNING and ERROR will be logged in the console log, errors additionally in the error.log file. Please make sure that TRACE or DEBUG severity levels are disabled in production environments, since they dramatically slow down the performance.

Logging is configured in the `<installation root>/configuration/hpm/log4j.xml` file.

2.4.8 Import

2.4.8.1 Preferences regarding CPU usage

Especially in Single Server scenarios in which the Import Jobs are executed throughout the day, it might be necessary to limit the CPU resource consumption of a single import process.

For this there are several preferences available which can be set in the `plugin_customization.ini` file.

```
# Specifies the maximum number of threads used for importing objects.
# Default is twice the number of cores.

# Allowed value is an integer greater than 0.

# com.heiler.ppm.importer5.core/importer.maxThreads=2

# Factor for the number of parallel threads.

# The number of parallel threads is computed by multiplying the number of cores
and this factor.

# The result is rounded.

# Value is a double; default is 2.0

# This value is only considered if importer.maxThreads is not set.

com.heiler.ppm.importer5.core/importer.parallelThreadsFactor = 2.0

# Allow to limit number of the import jobs running in parallel

# Only the actual database import (phase 2) is limited.

# -1 means unlimited (default)
```

```
com.heiler.ppm.importer5.core/importer.maxNumberOfDatabaseImportsAtSameTime =
-1
```

Additionally to these global settings, a user can activate the so called "debug mode" for the Import by pressing

Shift + F12 when **inside of the Import Perspective** in the Desktop Client. With activated debug mode, additional settings are available when scheduling the import.

Import

Schedule import

Specify when the import is to be executed.

As soon as possible ▾

Time ▾

Structure: Heiler Standard ▾

Import mode: Import or update all objects ▾

Error mode: Tolerant (object is imported as far as possible) ▾

☐ Perform test run before import, cancel import on errors

Test Settings

☐ Generate gold standard file

☐ Test against gold standard file

Number of threads:

☐ Use always bulkload

☐ Use never bulkload

Bulkload size:

☐ Clear catalog

Raw performance test (number of objects):

Storage mode for intermediate memory (0-2):

OK Cancel



Please do not change any of those additional settings except the **Number of Threads** !

2.4.8.2 Preferences regarding memory usage

When importing CSV or Excel files, per default all rows are read into heap memory in order to presort them. This speeds up processing if the data for one item is stored in rows that are spread. So when importing large CSV or Excel data files, consider to increase the job servers' heap size accordingly.

If presorting is not needed (because the data files are already presorted) or desired, it can be disabled by adding the following preference in the server's `plugin_customization.ini`:

```
com.heiler.ppm.mapper.core/presortRowsDefault = false
```



Importing large data files might become slower, when presorting is deactivated (see preference above) and the import data file is not already presorted.

2.5

Trouble Shooting

- Log files
- What's needed for a support call

2.5.1 Anti-Virus Software affecting performance

In the past we had some customer performance issues which were bound to the anti-virus scanning on the application server. Although we can not say that this is always the case for all deployments or all anti-virus products, we do recommend to exclude the PIM relevant directories from the scanning if applicable by the customers local security policy. If scanning can't be avoided in your situation, you should consider to reduce the lifetime of import or export jobs. The shorter they are, the less files will be in the import/export directories - which might increase scanning performance again.

2.5.2 Media Asset Provider (Classic Provider)

2.5.2.1 No preview are visible

Ensure following things:

1. Is the file extension unsupported?(E.g. pdf, ps, eps files because the ghostscript is not supported in PIM any more)
2. Is the graphicsMagick(official verified version GraphicsMagick-1.3.14-Q16) installed?
3. Check that the `blacklistExtensions` parameter of the `C:\heiler\server\configuration\HPM\plugin_customization.ini` file does not contain the file extension
4. Has the user who has started the application server write access to the media folder defined in the "Media Asset Server (MAS) Settings" section of the `server.properties` file?
Is it possible to open the image files from another computer by the explorer and another program?

5. Important: An unsuccessful rendering of a thumbnail leads to an entry in the database table MediaAssetFile" of the "[db_prefix]_MAIN_[db_suffix]" database.
Therefore, after a "subsequent" adjustment of these points this table must be modified. Set the bit-field "SupportThumbnail" to NULL.

Check 3rd party tools:

The technical background is that PIM - Server runs on a 64-bit machine as a 64-bit Java process.

PIM - Server starts `gm.exe` (Graphicsmagick) in a several process space (Graphicsmagick is a 32-bit process).

2.5.3 Obtain a Memory Dump (Heap Dump) from the Java Virtual Machine

There are several ways to obtain a heap dump during runtime.

- Heap dump on out of memory
By default all Product Manager launch configurations (development and runtime) contain the Java parameter `-XX:+HeapDumpOnOutOfMemoryError` which performs a heap dump when an out of memory error occurs.
- Heap dump on Ctrl+Break
Beginning with Java Version 1.5.0_16 the Java VM supports the `-XX:+HeapDumpOnCtrlBreak` parameter which will create a heap dump when you press Ctrl + Break in the Java console. For this the PIM - Server must be started with the `debug_console.cmd` file.
- Heap dump via jcmd utility
This console utility is included in JDK. Use `jcmd <processId> GC.heap_dump <filename>` for creating a heap dump, e.g. `jcmd 14904 GC.heap_dump D:\heapdump.hprof`
For details see <https://docs.oracle.com/javase/8/docs/technotes/guides/troubleshoot/tooldescr006.html>.

We suggest to use the Eclipse Memory Analyzer to analyze the Heap Dump, especially in case you have large heap sizes. It's the only tool we know so far which is able to open large heap dumps at all (you will need a lot of memory during the indexing of the heap dump in the memory analyzer - but this must not be physical memory 😊)

2.5.4 Enhanced Logging for Assortment Content Calculation

In case there are issues in terms of an assortment's content and there are items included/excluded which do not fit to the defined assortment rules, please add the following lines to the server's `log4j.xml` and execute an assortment update:

Assortment Log

```
<appender name="AssortmentFileAppender" class="org.apache.log4j.RollingFileAppender">
  <param name="File" value="logs/.assortment.log"/><!-- = Runtime directory -->
  <param name="MaxFileSize" value="10240KB"/> <!-- = 10MB -->
  <param name="MaxBackupIndex" value="10"/>
  <param name="Threshold" value="TRACE"/>
  <layout class="org.apache.log4j.PatternLayout">
    <!-- The default pattern: DateTime Priority [Thread] [Category] Message\n -->
```

```

    <param name="ConversionPattern" value="%d{ISO8601} %-5p [%t] [%c{1}] %m%n"/>
    <!-- Pattern to output the caller's file name and line number -->
    <!-- param name="ConversionPattern" value="%d{ISO8601} %-5p [%t] (%F:%L) -
    %m%n"/-->
    </layout>
</appender>

<!-- Default logging level for assortment modifying and updating.
This category uses its own appender and does not route log events to other appenders.
Use "TRACE" to activate assortment loggers -->
<category name="assortment" additivity="false">
    <priority value="TRACE"/>
    <appender-ref ref="AssortmentFileAppender"/>
</category>

```

The enhanced logging will write detailed messages regarding assortment content calculation to *.assortment.log* file located in the server's logging folder. Please provide this log file to Informatica support.

2.5.5 Server and client in different time zones

At the moment it is only possible to use P360 server and client in different time zones with a max time offset of 23 hours.

Theoretically it is possible to have a server client constellation with a 26 hour offset. This is not supported. More than 23 hours offset ends up in exceptions like

```

java.lang.IllegalArgumentException: Timestamp format must be yyyy-mm-dd
hh:mm:ss[.fffffffffff]
    at java.sql.Timestamp.valueOf(Timestamp.java:237)
    ...

```

2.5.6 Cluster

Since 8.1.1.04 there is a file located at `<sharedFolder>/shared/locks/datagrid/eventinfo.txt` which shows the current state of the cluster of Product 360 servers.

This file is useful for trouble shooting fail over issues.

2.5.7 Usage of strong cryptographic algorithms to encrypt/decrypt secure information

It is not longer necessary to enable the strong cryptographic algorithms manually. Unlimited cryptographic algorithms are enabled by default.

Nevertheless if you run into errors during encryption/decryption in Product 360, saying you're using an illegal key size you might need to enable the unlimited cryptographic algorithms manually.

This can be done by changing the configuration in file `<PIM`

`ROOT>\server\jre\lib\security\java.security`. Enable the property

`'crypto.policy=unlimited'` to activate the unlimited cryptographic algorithms.

This applies for all Modules of Product 360 running with JRE like Supplier Portal or Media Manager. If necessary please adjust the `jre\lib\security\java.security` of the corresponding Product 360 module.

2.6 DB Connection pool

PIM Core database workload consists of a large amount of parallel requests. To handle such workload PIM Core is using DB connection pool. Connection pool holds established jdbc connections and returns them to the pool when higher level data access layer such as Hibernate or UDA close them. Such pool helps to save time required to establish and prepare jdbc connection. However connection pool increase memory consumption. PIM Core maintains one connection pool per data source (MAIN,MASTER,SUPPLIER) and per data access module (UDA,Hibernate), total 6 connection pools. Quartz and JBPM maintain their own connection pools. Due to performance reasons PIM Core is using tomcatjdbc connection pool which is optimized for concurrent access. Popular implementations DBCP and c3p0 are not suitable for PIM Core workloads.

It is important to configure connection pool appropriately. See <https://tomcat.apache.org/tomcat-8.0-doc/jdbc-pool.html> for more detail on tomcat jdbc pool configuration options.

To improve jdbc performance the connection pool is capable of configuring prepared statement caches. This means prepared statements are cached per connection, and please note that this will significantly increase memory consumption! Also the connections need to be configured in a way that they do not remove idling connections too fast otherwise you will render the cache useless.

✓ JMX

Use hibernate and connection pool JMX managed beans to analyse workload and adjust values on production systems, see also [Tuning Advisory](#) (see page 56)

2.6.1 High performance connection pool configuration example

The default configuration should be sufficient for regular workloads. However in context of having many concurrent users and/or system jobs (import, export, data quality, etc.) the connection pool might become a bottle neck. The connection pool will throw an error in case the connections are exhausted, but this will not necessarily happen because there is a wait time until the error will be thrown. To adjust connection pool settings, the following data access template files must be adjusted.

❗ Oracle

When running PIM on Oracle do not enable caching of prepared statements unless you make sure that initialization parameter **OPEN_CURSOR** is set high enough. The limit is counted globally for all connections that belong to the same pool.

2.6.1.1 Hibernate

The template files when running PIM on Oracle for the Hibernate connection pools are located in the **server\configuration\HPM\database** folder:

- main.properties.template.ORA11g
- master.properties.template.ORA11g
- supplier.properties.template.ORA11g

In case you are running PIM on Microsoft SQL Server, the corresponding template files have the MSSQL2008 suffix.

Make sure to modify the template files since PIM will always generate the real property files during server startup.

✓ Configuration for 250 maximum active connections

```
# increase validation interval (value in milliseconds) to avoid permanent
connection check which is unnecessary in today's networks
hibernate.tomcatjdbc.pool.validationInterval = 120000

# increase eviction interval (value in milliseconds) when the pool should
check which idling connections could be removed from the pool
hibernate.tomcatjdbc.pool.timeBetweenEvictionRunsMillis = 30000

# increase idle time (value in milliseconds) before the connection may be
removed from the pool
hibernate.tomcatjdbc.pool.minEvictableIdleTimeMillis = 1800000

# increase the amount of initial connections since creating new
connections is a very time intensive task
hibernate.tomcatjdbc.pool.initialSize = 50

# increase the amount if minimum idling connections which will be kept
open at any time to fully utilize statement cache and avoid hard parsing
repeating queries
hibernate.tomcatjdbc.pool.minIdle = 125

# increase the amount of maximum idling connections to avoid closing
idling connections during peak times
hibernate.tomcatjdbc.pool.maxIdle = 250
```

```
# increase the amount of maximum idling connections to avoid the
situation that PIM processes are starving from available connections
hibernate.tomcatjdbc.pool.maxActive = 250

# lower the maximum wait time (value in milliseconds) until the
connection pool will throw an exception that the pool is exhausted
hibernate.tomcatjdbc.pool.maxWait = 10000

# enable jdbc interceptor for statement cache, increase max value if
required by application load pattern
hibernate.tomcatjdbc.pool.jdbcInterceptors =
ConnectionState;StatementFinalizer;StatementCache(prepared=true,callable=
false,max=250)

# Oracle only: this will force the jdbc driver to enable its builtin
statement cache to avoid hard parsing repeating queries
hibernate.tomcatjdbc.pool.connectionProperties =
oracle.jdbc.implicitStatementCacheSize=250

# Oracle only: prepared statements (INSERT or UPDATE) can be batched if
batch_size is > 1
hibernate.jdbc.batch_size = 50
```

All other settings should be left untouched. Increasing the amount of connections and/or statement cache size will result in higher memory consumption, so make sure that application server and database memory target is sized properly.

2.6.1.2 UDA

The template file when running PIM on Oracle for the UDA connection pools is located in the **server\configuration\HPM\database\uda** folder:

- uda-jdbcpool.xml.template.ORA11g

In case you are running PIM on Microsoft SQL Server, the corresponding template file has the MSSQL2008 suffix.

Make sure to modify the template file since PIM will always generate the real property files during server startup.

This example will adjust the same values like the Hibernate connection pool (please note that this is a xml file):

Configuration for 250 maximum active connections

```

<!-- increase validation interval (value in milliseconds) to avoid
permanent connection check which is unnecessary in today's networks -->
<property name="validationInterval">120000</property>

<!-- increase eviction interval (value in milliseconds) when the pool
should check which idling connections could be removed from the pool -->
<property name="timeBetweenEvictionRunsMillis">30000</property>

<!-- increase idle time (value in milliseconds) before the connection may
be removed from the pool -->
<property name="minEvictableIdleTimeMillis">1800000</property>

<!-- increase the amount of initial connections since creating new
connections is a very time intensive task -->
<property name="initialSize">50</property>

<!-- increase the amount if minimum idling connections which will be kept
open at any time to fully utilize statement cache and avoid hard parsing
repeating queries -->
<property name="minIdle">125</property>

<!-- increase the amount of maximum idling connections to avoid closing
idling connections during peak times -->
<property name="maxIdle">250</property>

<!-- increase the amount of maximum idling connections to avoid the
situation that PIM processes are starving from available connections -->
<property name="maxActive">250</property>

<!-- lower the maximum wait time (value in milliseconds) until the
connection pool will throw an exception that the pool is exhausted -->
<property name="maxWait">10000</property>

<!-- enable jdbc interceptor for statement cache, increase max value if
required by application load pattern -->
<property
name="jdbcInterceptors">ConnectionState;StatementFinalizer;StatementCache
(prepared=true,callable=false,max=250)</property>

<!-- Oracle only: this will force the jdbc driver to enable its builtin
statement cache to avoid hard parsing repeating queries -->
<property
name="connectionProperties">oracle.jdbc.implicitStatementCacheSize=250</
property>

```

Again, all other settings should be left untouched. Increasing the amount of connections and/or statement cache size will result in higher memory consumption, so make sure that application server and database memory target is sized properly.

2.7 Server Job Maintenance

Server jobs store their current state as well as all protocol entries in the database. A special job is executed once a day and is able to delete other, finished, jobs.


- [Configure the life time of a job \(see page 60\)](#)
- [Syntax \(see page 60\)](#)
 - [Example \(see page 60\)](#)
- [Standard Job List \(see page 60\)](#)

2.7.1 Configure the life time of a job

A job's life time is either defined as default life time by the job contribution itself (`plugin.xml`), or with the `plugin_customization.ini` file.

The values defined in the `plugin_customization.ini` will always have precedence over the default of the contribution.

To define a life time you need the unique job identifier (as listed in the table below), e.g. for the merge it's "Merge".

 A lifetime of 0 (zero) means no cleanup of the job, thus these jobs will not be removed from the job-history.

2.7.2 Syntax

```
com.heiler.ppm.jobhistory.cleanup.server/{JobTypeIdentifier}.lifeTime = {amount
in days}
```

2.7.2.1 Example

```
com.heiler.ppm.jobhistory.cleanup.server/Merge.lifeTime = 60
```

Means that already executed merge jobs are deleted after 60 days of their execution.

2.7.3 Standard Job List

List of standard jobs with their identifiers in order to configure their lifetime.

Job identifier	Job name	Job group	Default life time	Multiple Job entries
AssignDocumentJob	Media manager document assignment	System processes	3	TRUE
CheckTaskEscalatedJob	Check for escalated tasks	System processes	0	TRUE
CleanUpJobHistory	Remove obsolete processes	System processes	0	TRUE
CleanUpReports	Remove obsolete query results	System processes	0	FALSE
CleanUpSoftdelete	Remove deleted objects	System processes	0	FALSE
com.heiler.ppm.revision.jobType	Versioning	Data maintenance	10	TRUE
com.heiler.ppm.workflow.server.CleanUpFinished	Remove obsolete workflows	System processes	0	FALSE
CopyValueIntoStructureGroupsJob	Update structure group features	System processes	0	TRUE
create.thumbnails	Thumbnail generation	System processes	3	TRUE
create.thumbnails.rec	Recursive thumbnail generation	System processes	0	TRUE

Job identifier	Job name	Job group	Default life time	Multiple Job entries
DataQualitySeries	Data quality check (scheduled)	Data quality	0	TRUE
DictionarySynchronization	Dictionary synchronization (manual)	Data quality	0	TRUE
ExportDryRun	Quality check	Data maintenance	0	FALSE
ExportSeries	Export operations (repeated)	Publications	0	TRUE
FindReplace	Find and Replace	Data maintenance	2	TRUE
FindReplaceSetValue	Set	Data maintenance	2	TRUE
FulltextSearchJobSeries	Index creation (repeated)	Search index	30	TRUE
FulltextSearchSingleJob	Index creation (one-off)	Search index	30	TRUE
Import	Import operations	Data transfer	0	TRUE
Inbox	Hotfolder data processing	Data transfer	0	TRUE
InboxImport	Import operations by hotfolder	Data transfer	0	TRUE

Job identifier	Job name	Job group	Default life time	Multiple Job entries
ManualCleanUpReports	Remove obsolete query results (manual)	System processes	0	FALSE
MapArticlesToProductGroupJob	Product item feature transfer	System processes	0	TRUE
MapArticlesToStructureGroupJob	Update item-structure group assignments	System processes	0	TRUE
MapEGDsToStructureGroupJob	Update EGD-structure group assignments	System processes	0	TRUE
Merge	Merge operations	Data transfer	0	TRUE
MigrateStructures	Migrate Structures	Data maintenance	0	TRUE
NewMediaAssetDerivativeSchema	New derivative schema	System processes	0	TRUE
ReferenceDataDeployment	Deployment of reference data (scheduled)	Data quality	0	TRUE
SingleDataQuality	Data quality check (manual)	Data quality	0	FALSE
SingleExport	Export operations (one-off)	Publications	0	FALSE

Job identifier	Job name	Job group	Default life time	Multiple Job entries
SupplierExchangeExport	Export operations (Supplier Portal)	Publications	5	TRUE
TestExport	Test export operations	Publications	0	FALSE
update.version	Updating image versions	System processes	0	TRUE
UpdateTasksJob	Update tasks	System processes	0	TRUE

2.8 Soft Delete Cleanup Job

2.8.1 Why is there a concept of soft delete and a cleanup job that triggers the logging in the DB?

The soft delete mechanism allows to share information on deleted records in our DB with external systems using e.g. the deleted data providers in the Export. However, if you would keep all soft deleted objects in the DB forever it would lead to a DB space explosion and severe performance stretch and therefore a job can be configured to clean up soft deleted records from the DB after X days. A good thing to have for an enterprise application if you have a use case for this. If not, you may also want to switch the soft delete of completely and save some space on your DB.

Change the configuration in “plugin_customization.ini” and restart service

```
# Defines the default deletion mode in case entities support both, soft
and hard delete

# Supported values are SOFT and HARD

com.heiler.ppm.std.server/default.deletion-mode = HARD
```

Note: If the recovery logging level of your DB is set to FULL any change is stored in the transaction log for a roll back in case of errors. Otherwise you could only roll back using DB backups.

2.8.2 General information

The soft delete cleanup job is a maintenance job which performs a hard delete (physical delete) of entities which have been already soft deleted. The hard delete cleanup is only triggered for expired entities and only if the lifetime is not unlimited. With version higher 8.0.01, the lifetime is set to one year per default for all entities. The expiration date is calculated by means of the current date as basis. So if today is February the 5th and you have a lifetime of 1 month, then the calculated expire date will be January the 5th, which means that all entities deleted prior to this date will be removed permanently. The default life time property defines life time for all entity types. This can be overwritten per root and sub entity type.

2.8.3 Configuration

You can define the cleanup life time in the *plugin_customization.ini*. Search for `cleanup.lifeTime.default`.

Example default life times for all entities:

```
cleanup.lifeTime.default=1d
```

```
cleanup.lifeTime.default=1y
```

If you want to define a different life time for different root and sub entities then you need to uncomment the property `cleanup.lifeTime.custom.file` and refer to a custom file defining the soft delete properties. The custom file has to be located in the folder `~/server/configuration/HPM`. The `cleanup.lifeTime.custom.file` property referring to the custom file could look like this, for example:

```
cleanup.lifeTime.custom.file = softDeleteLifeTime.properties
```

In your customized soft delete life time properties file you can define the different life times for the entities. The most concrete definition will be used. So if you define a life time for a root entity, then you can overwrite it for the sub entity. But this is only valid if the life time for the sub entity is shorter than the one of the parent.

A valid configuration:

```
cleanup.lifeTime.default = unlimited
```

```
ArticleType.lifeTime = 1y
```

```
ArticleLangType.lifeTime = 1m
```

```
StructureGroupLangType = 6d
```

2.8.4 Schedule

To schedule the job repetitively, uncomment the property `cleanup.job.repeatPattern` and define the pattern. From the version 8.0.01.00 this job is scheduled to run every day at 24:00. To disable the scheduling, you have to uncomment the property and leave it empty, then the job won't be executed automatically but may

be started manually in the process overview (Select "System processes > Remove deleted objects", select the process in the "Current and past processes" view and call "Execute immediately" from the context menu).

Example for running the job daily at 24.00:

```
cleanup.job.repeatPattern = 0 00 00 * * ? *
```

To disable scheduling:

```
cleanup.job.repeatPattern =
```

For more information refer to the Quartz Enterprise Job Scheduler documentation for Cron Expressions

2.8.5 Be cautious with mass deletes in such a setup

The recovery level FULL would increase your transaction log if your DBA is not securing the log away regularly. Potentially endless so to speak, or just until the disk space runs full. A one-time mass deletion of records in Product 360 under recovery level FULL may cause such problem once the cleanup job kicks in.

Recovery level SIMPLE for example would delete the log after every commit to free up log space again.

2.8.5.1 How do other customers solve such a constellation?

So far, we have not seen many customers deleting that many records in such a short period of time. However, we have seen 4 different variations on best practices in this scenario:

1. They give the DB as much space as it would need. Even temporary...
2. They have the cleanup job executed frequently in general (typically delete everything older than 4 weeks) to keep the overall amount of soft deleted records as low as possible over time
3. In case of a one-time mass deletion process customers create a DB backup and then set the logging level to SIMPLE for the time needed by the transaction
4. They back up the transaction log prior this action to start with an empty log and keep disk space consumption as limited as possible

These are scenarios where it is already identified that this mass deletion could otherwise create a problem. However, what happens if you didn't know about this and now are stuck with a transaction log running full all the time the cleanup job gets executed?

2.8.5.2 Possible Workarounds

Set the X days a record should be kept in the DB relatively high to reduce the amount of overall records in the queue to be cleaned up as small as possible per job execution. This needs some further monitoring over time but is the best option in a safe setup to reduce the amount of soft deleted records controlled and as granular as possible.

The key question is whether you have a lot of soft deleted records in the queue (e.g. because a clean-up job has only recently been configured for the first time in your setup) or if that space problem is caused only having deleted records in the queue that share the same day on their deletion time stamp. If the latter is the

case you could still try to work with the bulk size property to get things solved by defining the number of records to be taken by the job per transaction:



Add this line to the plugin_customization.ini of the server to limit the number of items per transaction to 1000 (or whatever suits your idea, default is 100,000). Better use control center to adjust the file and deploy it on the servers if you have multiple servers.
com.heiler.ppm.std.server/entity-manager.delete.bulk-size = 1000

2.9 Data Quality Operation

2.9.1 Logging

Additional logging information for data quality execution can be increased on several levels: rule engine execution, data provisioning and performance analysis

2.9.1.1 Rule engine execution

In order to see logging information of the execution of a data quality rule by the rule engine set the category 'IDQ' to DEBUG in server log4j.xml

```
<category name="IDQ">
  <priority value="DEBUG"/>
</category>
```

As opposed to normal logging information a server restart is required for changes to take effect. The logs can be found in separate files in the server logs folder. Each of them has a pattern like <Execution Thread name>-<Executed rule name>-<Execution start time>.log

2.9.1.2 Data provisioning

In order to see logging information of the data provisioning (data input into and data output from rule) set the category 'DQ_EXECUTION' to DEBUG in server log4j.xml.

```
<category name="DQ_EXECUTION">
  <priority value="DEBUG"/>
</category>
```

The output is available in the usual error.log respectively console.log files. Example output:

```
13:01:25,166 DEBUG [exe-IDQ-SDK-Check_PresetValues-2] [DQ_EXECUTION] DQ IN line: 0
(rule: preset violation)
inObjectID: '33762@1' (field: Object_ID)
```

```

inAttributeName: 'Color' (field: ArticleAttribute:name)
inLanguage: 'English' (field: ArticleAttributeLang:language)
inStructure: '10058' (field: ArticleAttributeStructureGroupAttributeMap:structure)
inStructureGroup: '232@10058' (field:
ArticleAttributeStructureGroupAttributeMap:structureGroup)
inStructureFeature: '74@10058' (field:
ArticleAttributeStructureGroupAttributeMap.StructureAttribute)
inAttributeValue: 'green2' (field: ArticleAttributeValue.Value)
inValueIdentifier: 'DEFAULT' (field: ArticleAttributeValue:identifier)
inStructurePresetValue: '' (field:
ArticleAttributeStructureGroupAttributeMap.DomainValue)
inDataType: 'Character string' (field: ArticleAttribute.Datatype)

13:01:25,317 DEBUG [exe-IDQ-SDK-Check_PresetValues-1] [DQ_EXECUTION] DQ OUT line: 0
(rule: preset violation)
status_code: '0' (field: QualityStatusEntry.Status)
status_message: 'Primary, Pflichtgruppe, Feature 'Color': Value 'green2' (English,
DEFAULT) of attribute 'Color' does not conform to the preset values' (field:
QualityStatusEntry.Message)
object_id: '33762@1' (field: Object_ID)

```

2.9.1.3 Performance analyzing

Enable the following category to DEBUG for additional information about DQ initialization parameters and DQ Executor Pool usage.

```
<category name="com.heiler.ppm.dataquality.server">
  <priority value="DEBUG"/>
</category>
```

2.9.2 Performance tuning

- Increase number of maximum jobs executed in parallel by adjusting the setting **com.heiler.ppm.job.server/maxRunningServerJobs** accordingly.
- Each pool properties are available via JMX and also the size of each DQ Executor Pool can be adjusted at runtime via JMX (MBeans → com.heiler.ppm → dataQuality → dqExecutorPools)

[illegible]

- Use synchronous execution of DQ via REST API instead of asynchronous jobs and waiting for job finished result (e.g. when executing rules inside a bpm workflow).
This decreases the general load of the job framework and avoids the general overhead of job scheduling; see also: Rest Management API -> Rest Data Quality API -> Rule Execution.

- The precision of the mapping input ports should be sized as small as reasonably possible, since this has a direct impact on rule execution performance and DTM memory allocation
- Rule execution performance could benefit from setting a lower value for **com.heiler.ppm.dataquality.server/dataquality.executor.pageSize** as the data sets that are needed to be retrieved will be smaller, hence faster
- Also consider having a smaller value for **com.heiler.ppm.dataquality.server/dataquality.executor.pageSize** in general since the executor can then be used earlier in a different execution scenario while the system is busy writing the result of the current elements of the DQ Execution Result to the database (the executor pool serves FIFO)
- Rule execution performance could benefit from setting a higher value for **com.heiler.ppm.dataquality.server/dataquality.execution.dtmMaxMemPerRequest** to avoid that DTM has to swap out cache and index files to disk during rule execution, however this is subject to the available physical memory of the host. An alternative could be moving the temporary DQ folder to a RAMDisk by setting **com.heiler.ppm.dataquality.server/dataquality.execution.tmpFileLocation**
- Depending on the hardware the setting **com.heiler.ppm.dataquality.server/dataquality.execution.maxThreads** can be increased to utilize the available CPU cores

The following properties in the preferences.ini show default values and may be adjusted to improve performance depending on corresponding use-cases:

```
# Enables an alternative implementation for the DQ Rule Check_IsEmpty that doesn't
use the IDQ SDK.
# Set to true in case you want to avoid DTM overhead for simple isEmpty checks.
# Default: false
com.heiler.ppm.dataquality.server/dataquality.alternativeCheckIsEmpty=false

# Specifies the page size when loading the list model of data to be checked for IDQ
rule execution.
# Default: 10000 objects
com.heiler.ppm.dataquality.server/dataquality.executor.pageSize=10000

# Maximum amount of concurrent executors in the executor pool.
# Each executor can run independent from each other to avoid contention on specific
IDQ rules.
# Default: 2
com.heiler.ppm.dataquality.server/dataquality.executionPool.maxTotalPerRule=2

# Example: maximum amount of concurrent executors in the executor pool per specific
IDQ rule.
# The '/' in the rule identifier has to be replaced with '_'
com.heiler.ppm.dataquality.server/
dataquality.executionPool.maxTotalPerRule.Informatica_PIM_Content_PreDefined_Rules_Ch
eck_DataTypes=5

# Specifies in seconds how long the MappingExecutor instances will be kept in the
pool before aging out.
# Exceptions:
# - Executors will be destroyed immediately in case a new version of a rule will be
uploaded
# - Executors will be destroyed immediately when related dictionaries have been
changed
```

```

# - Executors can only age out without a dictionary change in case they have been
used at least once
# Default: 3600 seconds (60 minutes)
com.heiler.ppm.dataquality.server/
dataquality.executionPool.executorKeepAliveInterval=3600

# Example: specifies in seconds how long the MappingExecutor instances will be kept
in the pool before aging out per specific IDQ rule.
# The '/' in the rule identifier has to be replaced with '_'
com.heiler.ppm.dataquality.server/
dataquality.executionPool.executorKeepAliveInterval.Informatica_PIM_Content_PreDefine
d_Rules_Check_DataTypes=14400

# Specifies the amount in bytes the DTM should be allocating from the host OS memory
(note that this is not JVM heap memory).
# If configured properly this will avoid swapping cache files to disk during
transformations.
# Default: empty (DTM will automatically determine a value)
dataquality.execution.dtmMaxMemPerRequest=

# Specifies the amount in bytes the DTM should be allocating from the host OS memory
per specific IDQ rule.
# The '/' in the rule identifier has to be replaced with '_'
#dataquality.execution.dtmMaxMemPerRequest.Informatica_PIM_Content_PreDefined_Rules_C
heck_DataTypes=58720256

# Specifies the amount of threads in the IDQ SDK executor thread pool resp. how many
executors can run at the same time.
# This roughly translates to dedicated CPU cores for IDQ execution.
# Default: 5
com.heiler.ppm.dataquality.server/dataquality.execution.maxThreads=5

# Specifies the location for the temporary files for each executor that are created
during IDQ execution.
# Default: empty (using 'workspace/.metadata/com.heiler.ppm.dataquality.server' as
default)
dataquality.execution.tmpFileLocation=

```

An additional note to the property **dataquality.executionPool.maxTotalPerRule**: all rule executors will be already created during server startup in background threads.

2.9.2.1 Cleanup of Status Entries

When executing data quality, the execution result is written to the database table StatusEntry, referring to the used executed data quality rule configuration.

If a data quality rule configuration is deleted (e.g. via Desktop Client DQ Perspective), the status entries for that deleted configuration are not deleted immediately.

A cleanup job will handle this, which runs every day at midnight per default.

The identifier of that job is 'CleanUpDataQualityStatusEntries' and the (english) name 'Status entries cleanup (scheduled)'.

plugin_customization.ini

```
com.heiler.ppm.dataquality.server/dataquality.status.cleanup.job.pattern = 0 0 0 ? *
1,2,3,4,5,6,7
```

Additionally, it is recommended to activate soft deletion of status entries. To do this, the already existing soft delete job has to be used.

Please refer to the 'Soft Delete Cleanup Job' chapter in the 'Server Operation' documentation and apply it to the sub entity type 'StatusEntryType'. Also ensure, to configure the lifetime for the root entity type 'StatusType' to another value than "unlimited" (e.g. '1y'), and be aware, that thereby also other sub entity types of 'StatusType' (e.g. 'PublicationStatusEntryType', which is used by the GDSN Accelerator) will also be cleaned up.

This is because the most concrete definition for an entity will be used. So if you define a life time for a root entity, then you can overwrite it for the sub entity. But this is only valid if the life time for the sub entity is shorter than the one of the parent.

2.9.3 Issue reporting guide

This guide explains required information and steps to retrieve them in order to ensure a certain overall quality level for issues concerning Data Quality relevant topics.

Please provide as much information as described as follows in order to have a quick success in solving the issues.

2.9.3.1 Configuration setup

All DQ relevant configuration and rule files contained in the server configuration subfolder 'dataquality'. This can simply be done by zipping the whole folder and attaching it to the issue.

Screenshots of specific rule configurations are also helpful, especially when they are custom ones. The same is true for the configured execution triggers.

It may also important to provide information about the setup workflows if executions are called from there.

2.9.3.2 Information about the execution context

Data Quality executions have following main informations

- The rule configurations containing information about the data provisioning
- The actual data quality rules that are used in the rule configuration
- The execution may contain more than one rule configuration, mostly if the execution is done for specific channels containing several configurations
 - If the rule configurations are executed as jobs as f.g. by manual executions, see the job log details in the processes overview for the used rule configurations.
 - For triggered executions this information may also be retrieved by looking at the trigger configuration
 - Execution may be also triggered from calls inside workflows

- Can the problem be pinned down to a specific rule configuration or rule? If so, Please specify them. Is the rule a custom rule and if so, what is it supposed to do?

2.9.3.3 Logging data

- General logging data as always (error.log, console.log)
- DQ specific logging data as described in the previous 'Logging' section. This may require to change the logging level to DEBUG as described there and then execute the rules again in order to see the additional logging information.

2.9.3.4 Performance

- Provide information about the settings as described in the Performance tuning section
- Is the execution manually or triggered e.g. after import or every night or for each item change?
- How many items are affected? Execution for single item or many items of a catalog or other data set f.g.?
- Are there many executions in parallel?

2.9.3.5 Crashes

In case of crashes get the heap and thread dump of the P360 Server.

If the crash happens on the DQ engine itself, also provide heap dumps of the underlying engine shared libraries.

For Windows this can be done like this:

- ensure
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps`
 registry key is present
- try to reproduce the problem. Dumps are available in
`%LOCALAPPDATA%\CrashDumps`
 or
`C:\Windows\System32\config\systemprofile\AppData\Local\CrashDumps`
 in case service is run with system account.

2.9.4 Configure rules that are affected by Dictionary Synchronization in order to increase performance of IDQ rule execution

2.9.4.1 Situation/ motivation

- For each IDQ rule, PIM creates executor objects and stores them in a pool, because creation is expensive. Executors are used to execute IDQ rules in parallel. These executors are created lazily on demand. Also IDQ SDK internally synchronizes executor creation, which means, creation is queued.
- Changes to dictionaries aka reference tables are only picked up by the rule if the executor is disposed and re-created

- Currently, on the PIM side, it is not known which reference tables are used by which IDQ rule. Hence, whenever, a dictionary is synchronized, all executors are disposed, and created lazily on demand.
- Because of that the performance drastically degrades whenever a dictionary is synchronized AND many new executors are requested for IDQ rule execution.

2.9.4.2 Solution

- It is now possible, to configure, which IDQ rules are affected by which dictionary (aka Reference Table). Therefor the `DictionaryRuleConfiguration.xml` is used, which is stored in the PIM server's folder `/configuration/HPM/dataquality`, and looks like this:

DictionaryRuleConfiguration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<dictionaryToRuleConfiguration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="DictionaryRuleConfiguration.xsd">
  <isActive>true</isActive>
  <dictionary>
    <dictionaryIdentifier>Informatica_DQ_Content\Dictionaries\General\profanity_infa.
dic</dictionaryIdentifier>
    <affectedRules>
      <ruleIdentifier>Informatica_PIM_Content/PreDefined_Rules/Check_Profanity</
ruleIdentifier>
      <ruleIdentifier>Informatica_PIM_Content/Custom_Rules/MyCustomRule1</
ruleIdentifier>
    </affectedRules>
  </dictionary>
</dictionaryToRuleConfiguration>
```

- This configuration is only active, if this XML file is placed in the mentioned folder AND if `<isActive>` is set to `true`. Otherwise, the behavior is the same as before, means with every dictionary synchronization ALL rule executor objects are disposed and created lazily on demand.
- The `<dictionary>` element defines, which rules (defined by the `<ruleIdentifier>`s within the `<affectedRule>` element) are affected by a specific dictionary (specified by the `<dictionaryIdentifier>`). You can define as many `<dictionary>` elements as you wish.
- The `<dictionaryIdentifier>` has to be copied from the dictionary view (part of the Dictionary perspective) in the PIM Desktop client.
- The `<ruleIdentifier>` is composed of the folders and subfolders of the IDQ rule and the name of the IDQ rule itself. This information can be found through the Data quality perspective in the PIM Desktop Client:
- When a dictionary, which is configured here, is synchronized, ONLY the executor objects of the configured affected rules are destroyed and recreated lazily on demand - all other rule executor objects are not hit.
- When a dictionary, which is NOT configured here, is synchronized, the behavior is the same as before, means ALL rule executor objects are disposed and created lazily on demand.



Be aware, that as soon as a specific dictionary is configured in the `DictionaryRuleConfiguration.xml` file, synchronization of that dictionary NEVER leads to a destruction of not affected rule executor objects. That means that if those "not affected" rules executor objects might anyhow have dependencies to the configured dictionary, they might work on a very old state of this dictionary, which might lead to an inconsistent behavior when executing IDQ rules.



The **DictionaryRuleConfiguration.xml** configuration is only loaded once during server startup. Means, after changing that file, the PIM server has to be restarted, in order to activate the changes.

If this something goes wrong, during loading of this configuration file (e.g. the corresponding **DictionaryRuleConfiguration.xsd** is not existing in the `/configuration/HPM/dataquality` folder, or there is an syntax error in the **DictionaryRuleConfiguration.xml** file), an ERROR is logged in the server's log file. But still the server starts up and the behavior is the same as before, means with every dictionary synchronization ALL rule executor objects are disposed and created lazily on demand.

2.9.5 Status Cache Initialization

- [Overview](#) (see page 74)
- [Schema Changes](#) (see page 74)
- [Initialization](#) (see page 75)
 - [Database](#) (see page 75)
 - [Network](#) (see page 75)
 - [Local Storage](#) (see page 76)

2.9.5.1 Overview

The Status Cache is responsible for providing the execution result for every Data Quality execution for each corresponding entity object.

Instead of loading each result from the database, all results are stored in-memory.

The initialization of the Status Cache takes place during server startup because vital functionality depends on it (e.g. the Data Quality Dashboard).

2.9.5.2 Schema Changes

The StatusRevision table has a new column called StatusEntryJSON which contains a binary compressed JSON object which holds all StatusEntry records.

The application server checks for the existence of the that column, and in case it does not exist it will create and populate it during server startup.

2.9.5.3 Initialization

The StatusCache initialization supports multiple strategies for populating the cache:

- Initialization from Database
- Initialization over Network
- Initialization from Local Storage

Both Network and Local Storage strategies are optional and can be disabled, however they offer superior performance.

Database

The initialization from database supports parallel processing. By default, parallel processing is enabled if the database contains more than 200,000 status objects.

The default parallel degree is determined upon the db.available.cpu setting in the server.properties file.

Database-related properties

```
# Status cache initialization will utilize multiple threads once the threshold has
# been reached.
# Default: 200000
com.heiler.ppm.status.server/statusCache.parallelTreshold = 200000

# The maximum amount of concurrent threads to query the database.
# Default: empty (using "number of DB CPU cores" configured in server.properties)
com.heiler.ppm.status.server/statusCache.parallelDegree =
```

Network

The initialization over network supports parallel processing and is enabled by default.

Since there is only a single socket connection between servers, the idea behind the parallel degree is that one thread is compressing the cache elements while another thread is sending the data to the other server.

Network-related properties

```
# Allows the status cache initialization over network.
# Default: true
com.heiler.ppm.status.server/statusCache.networkEnabled = true

# The amount of status cache elements contained in every network request.
# Default: 500000
com.heiler.ppm.status.server/statusCache.networkBatchSize = 500000

# The maximum amount of parallel threads to transfer the status cache over the
# network.
```

```
# While there can be only one active socket at a time, the other threads will be
preparing their payload.
# In case there are less CPU cores available, all CPU cores except one will be
utilized.
# Default: 4
com.heiler.ppm.status.server/statusCache.networkParallelDegree = 4
```

Local Storage

The initialization from local storage supports parallel processing and is enabled by default. It is by far the fastest initialization strategy.

During shutdown, the application server will save its current StatusCache instance like a snapshot to the local storage, using a compressed LZF binary format.

The application server will use all available CPU cores except for one during the shutdown process, because saving the snapshot is a CPU bound process due to the compression algorithm. Likewise, the application server will write one data file per available CPU core.

Then during startup, the server reads the timestamp of the snapshot and performs a clean refresh of all changed status objects from the database, using the snapshot timestamp.

Since the local storage has a hard I/O limit, more reader threads do not necessarily mean better performance - actually it's quite the opposite.

This is why the default is only uses four CPU cores and not all available CPU cores.

Local Storage-related properties

```
# Allows the status cache initialization from local storage.
# Default: true
com.heiler.ppm.status.server/statusCache.localStorageEnabled = true

# The local file path of the status cache snapshot
# Default: empty (using workspace)
com.heiler.ppm.status.server/statusCache.localStoragePath =

# The maximum amount of parallel threads to read the status cache from the local
storage.
# This process is I/O bound, i.e. increasing parallel degree could lead to worse
performance.
# In case there are less CPU cores available, all CPU cores except one will be
utilized.
# Default: 4
com.heiler.ppm.status.server/statusCache.localStorageParallelDegree = 4
```

2.10 Check integrity of attribute names in key language

The name of an attribute must exist in the key language otherwise errors may occur during operation. The key language can be specified in the server.properties.

If the key language is changed, it is possible that attributes exist that do not have a name in the key language. The following database script can be used to check if all attributes have a name in the key language.

2.10.1 MSSQL

The script for MASTER and SUPPLIER

```
SELECT COUNT(aa1.ID)
FROM [ArticleAttribute] aa1
INNER JOIN [ArticleAttributeLang] aal1 ON aa1.ID = aal1.[ArticleAttributeID]
WHERE aa1.ID NOT IN (
    SELECT aa2.ID
    FROM [ArticleAttribute] aa2
    INNER JOIN [ArticleAttributeLang] aal2 ON aa2.ID = aal2.[ArticleAttributeID]
    WHERE aal2.[LanguageID] = :keyLanguageID )
```

The script for MAIN

```
SELECT COUNT(sga1.ID)
FROM [StructureGroupAttribute] sga1
INNER JOIN [StructureGroupAttributeLang] sgal1 ON sga1.ID = sgal1.
[StructureGroupAttributeID]
WHERE sga1.ID NOT IN (
    SELECT sga2.ID
    FROM [StructureAttribute] sga2
    INNER JOIN [StructureGroupAttributeLang] sgal2 ON sga2.ID = sgal2.
[StructureGroupAttributeID]
    WHERE sgal2.[LanguageID] = 9 )
```

2.10.2 Oracle

The script for MASTER and SUPPLIER

```
SELECT COUNT(aa1.ID)
FROM "ArticleAttribute" aa1
INNER JOIN "ArticleAttributeLang" aal1 ON aa1.ID = aal1."ArticleAttributeID"
WHERE aa1.ID NOT IN (
    SELECT aa2.ID
    FROM "ArticleAttribute" aa2
    INNER JOIN "ArticleAttributeLang" aal2 ON aa2.ID = aal2."ArticleAttributeID"
    WHERE aal2."LanguageID" = :keyLanguageID )
```

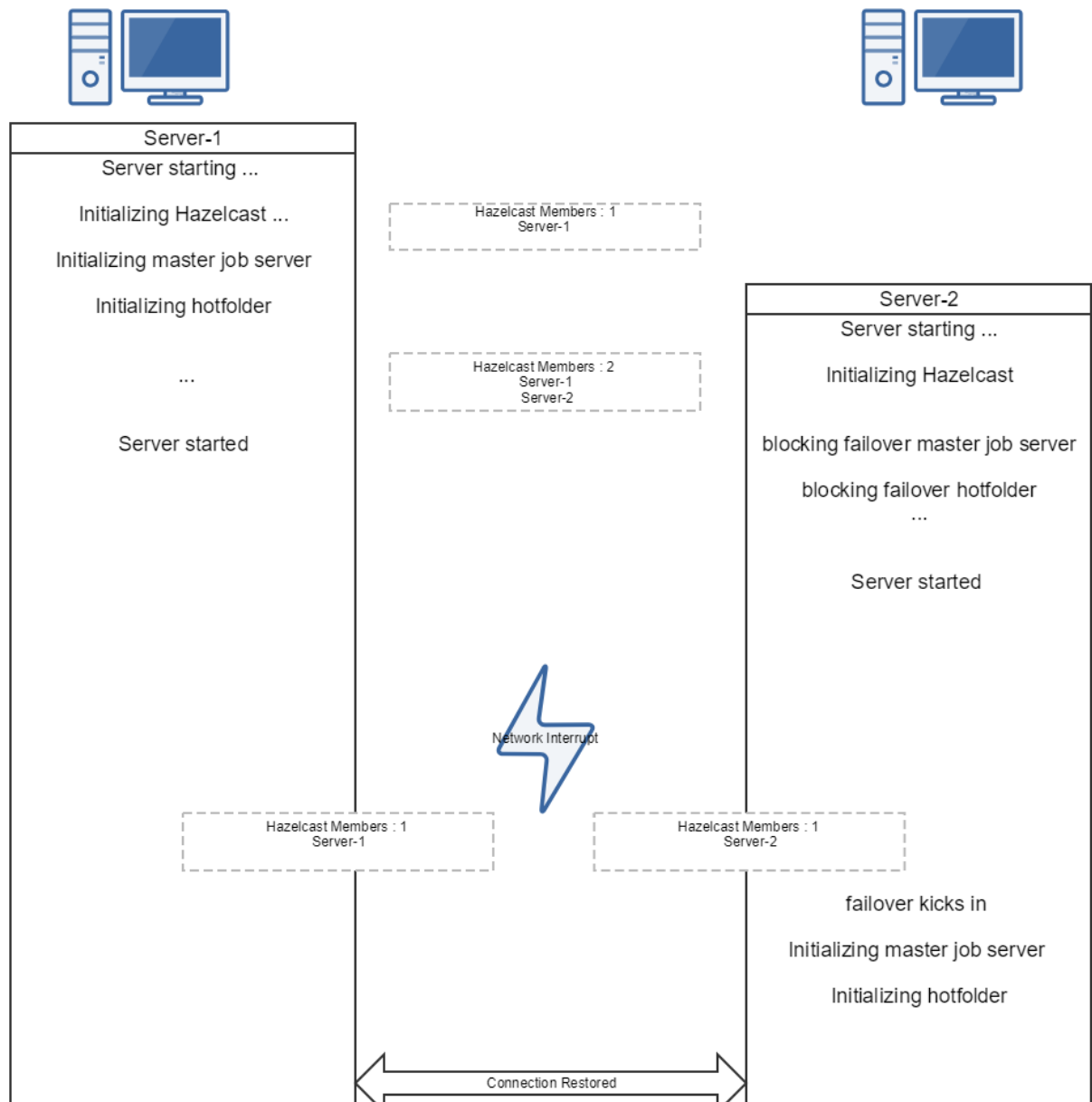
The script for MAIN

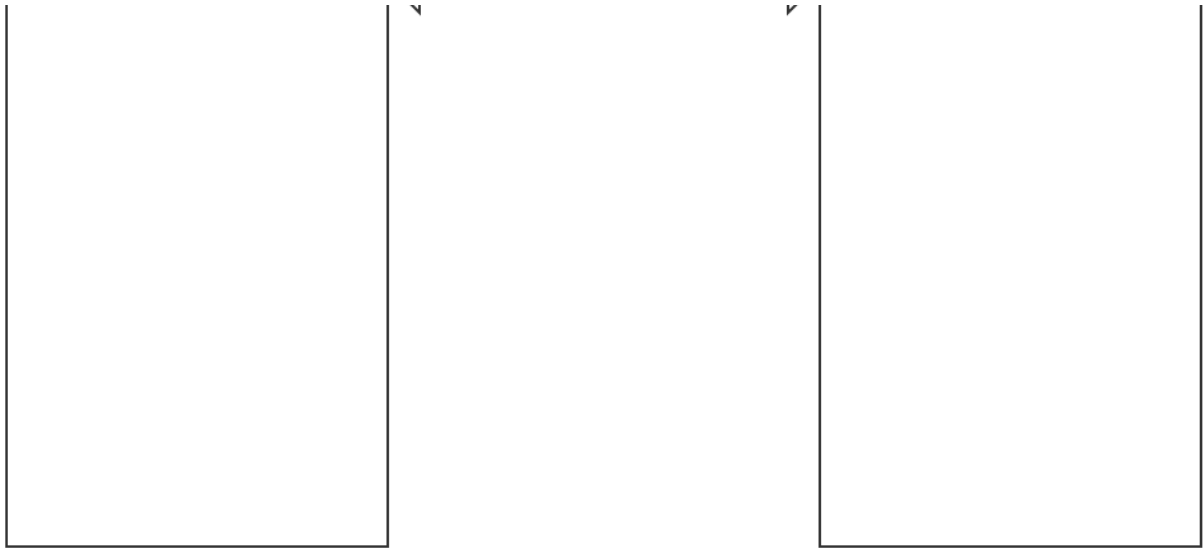
```
SELECT COUNT(sga1.ID)
```

```

FROM "StructureGroupAttribute" sga1
INNER JOIN "StructureGroupAttributeLang" sga1l ON sga1.ID = sga1l."StructureGroupAttributeID"
WHERE sga1.ID NOT IN (
    SELECT sga2.ID
    FROM "StructureAttribute" sga2
    INNER JOIN "StructureGroupAttributeLang" sga2l ON sga2.ID = sga2l."StructureGroupAttributeID"
    WHERE sga2l."LanguageID" = 9 )
    
```

2.11 Network Recovery





2.12 Monitoring with Micrometer

2.12.1 General monitoring approach

The Product 360 server exposes several metrics leveraging the micrometer library (<https://micrometer.io/>). These monitoring systems are supported by the Product 360 server:

- CloudWatch
- Graphite
- NewRelic
- Dynatrace
- Elastic
- Prometheus

Configure the monitoring system you want to use in the micrometer.properties file located in **<PIM_SERVER_INSTALLATION_ROOT>\server\configuration\HPM\micrometer.properties** of the server installation package to expose metrics. The application server needs to be restarted in order to have changes take effect. A list of JVM default metrics will be exposed as well as Product 360 custom metrics. Find a list of the available custom metrics below.

2.12.2 List of Product 360 custom metrics

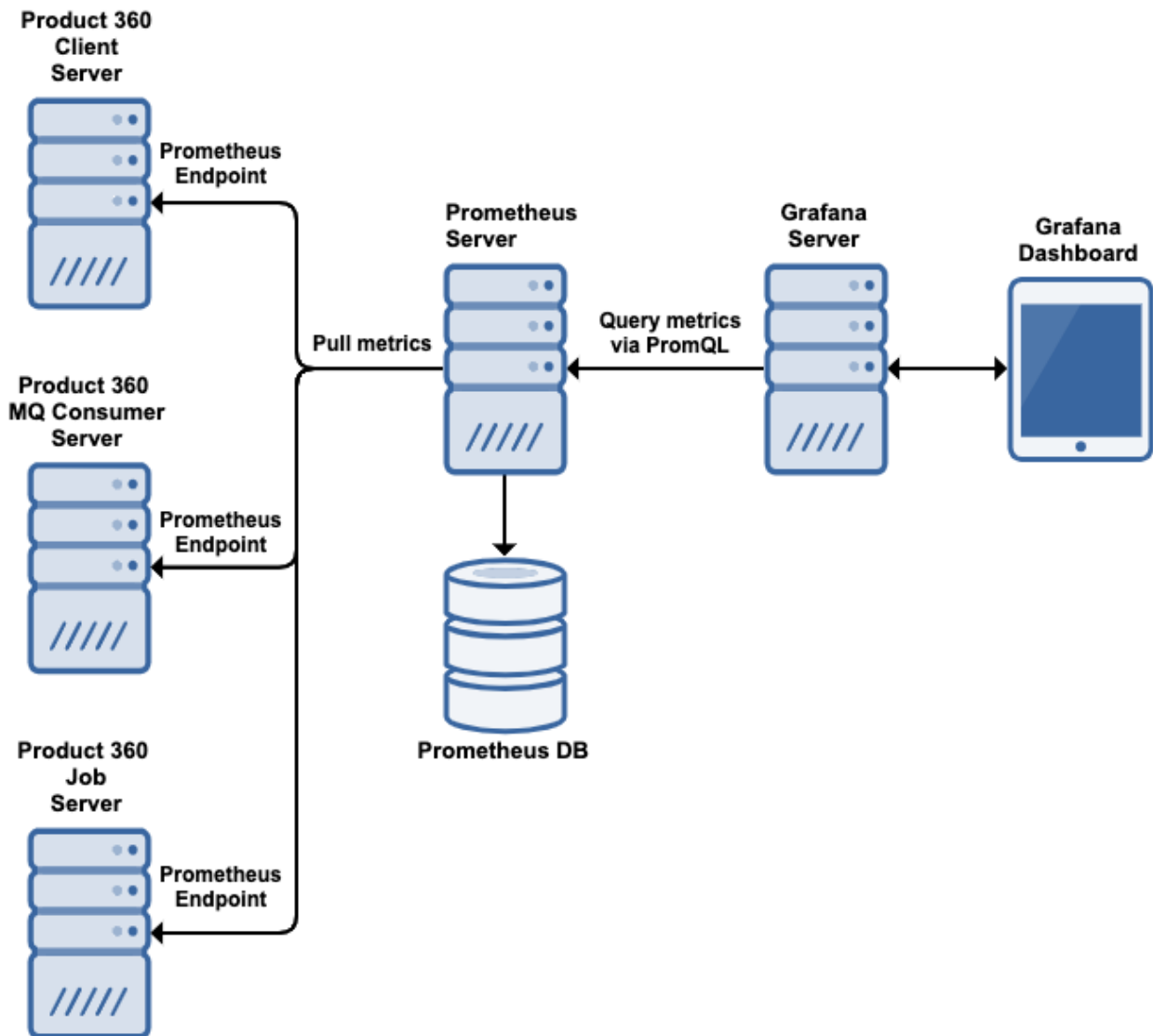
Meter name in prometheus endpoint	Tags	Description
queue_messages_read_total	queueName - Name of the monitored queue	Number of messages this Product 360 server has read from the queue with queueName

Meter name in prometheus endpoint	Tags	Description
queue_messages_read_success_total	queueName - Name of the monitored queue	Number of messages this Product 360 server has successfully processed after it read them from the queue with queueName
queue_messages_read_error_total	queueName - Name of the monitored queue	Number of messages this Product 360 server has read from the queue with queueName, but errors occurred when they were processed
queue_messages_write_total	queueName - Name of the monitored queue	Number of messages this Product 360 server has written to the queue with queueName
queue_messages_write_error_total	queueName - Name of the monitored queue	Number of messages this Product 360 server has tried to write to the queue with queueName, but failed to do so

2.12.3

Sample for monitoring a Product 360 server with Micrometer, Prometheus and Grafana

Architecture overview to monitor Product 360 servers with Micrometer, Prometheus and Grafana.



The following steps are required to enable monitoring of Product 360 server with Micrometer, Prometheus and Grafana.

- Enable the Prometheus endpoint for the Product 360 server you want to monitor by adjusting the values in the file micrometer.properties:

✓ **Micrometer settings in micrometer.properties**

```
#####
####
### Prometheus settings
###
#####
####
prometheus.enabled           = true
prometheus.uriPath           = /prometheus
```

```
prometheus.port           = 9090
prometheus.descriptions   = true
#prometheus.step           =
```

- As a minimum enable prometheus, set the uriPath and the port number
- Restart the Product 360 server
- Verify metrics are available by accessing the Prometheus endpoint you specified. In this sample `http://<your_server_hostname>:9090/prometheus`
- Use a Prometheus server to pull the metrics from the Product 360 endpoints and to write them into a database. You need to add the Product 360 Prometheus endpoint as a scrape target to the `prometheus.yml` file. For details check the Prometheus configuration documentation
- Use a Grafana server to visualize the collected metrics data in Prometheus. Therefore Prometheus is added as data source to Grafana and the meters are queried using PromQL (Prometheus Query Language). To visualize the JVM default metrics you can import one of the available Grafana template dashboards as for instance: `https://grafana.com/grafana/dashboards/4701` To visualize the Product 360 custom metrics you can build your own dashboard and query them by the names listed in the custom metrics table or you can import the Product 360 dashboard containing the Product 360 custom metrics.
`Product_360_Grafana_Dashboard.json` Specify your Prometheus server as data source during import.

2.12.4 Metrics and Kibana Dashboard

With the inbuilt Micrometer instrumentation façade, Product 360 10.1 delivers over 150 system and application-specific dimensional metrics for actionable health insights and reporting of the environment. With vendor-neutral meters, it is now possible to independently provision an application performance monitoring system of your choice. Additionally, Product 360 10.1 ships with an accelerator that includes predefined Elastic Kibana based reference dashboards providing an overview of all key metrics for application monitoring - ready to use and adopt as needed.

The purpose of this document is to guide the user regarding the installation, configuration and usage of the Elastic Kibana based reference dashboards.

2.12.4.1 Installation, Configuration and Operation



Installation





Accelerator package

The application monitoring dashboards are available as out-of-the-box visualizations, in JSON file format, which can be imported directly through the Kibana UI.

The JSON files and reference configurations can be found in the within the **PIM_10.1.0.00.00_MicrometerDashboard** directory in the **PIM_10.1.0.00.00_Accelerators.zip**

Once downloaded, extract the zip to a folder. Preferably Informatica folder for ease of access. The extracted folder would look as below:

<input type="checkbox"/> Name	Size
 P360 Micrometer Kibana Dashboard Config	
 Recommended-log4j2-Micrometer.xml	4 KB

<input type="checkbox"/> Name	Size	Type
 p360_advanced_settings.ndjson	1 KB	NDJSON File
 p360_kibana_dashboard.ndjson	712 KB	NDJSON File
 p360_kibana_index.ndjson	16 KB	NDJSON File
 p360_kibana_visualization.ndjson	585 KB	NDJSON File

Configuration

ElasticSearch Configuration

The ElasticSearch configuration has to be done in the **micrometer.properties** file of the corresponding application server. Scroll down to the section for Elastic search and ensure the settings as shown below .

micrometer.properties

```
#####
### Elastic settings                                     ###
#####
elastic.enabled = true
# Ensure that access credentials for elastic search host are added correctly
# Please note to not have a trailing slash at the end of the host name!
elastic.host = http://myElasticServer.myCompany.com:9200
elastic.userName = MyUser
elastic.password = MyUserPassword
#Do not change the name of the index, otherwise the provided Kibana visualisations
#will also need to be adjusted as the index name is configured there!
elastic.index = p360_metrics
elastic.indexDateFormat = yyyy-MM
elastic.timestampFieldName = @timestamp
elastic.autoCreateIndex = true
elastic.step = 1m

# Optional parameters
```

```
#elastic.pipeline           =
#elastic.indexDateSeparator =
#elastic.connectTimeout    =
#elastic.readTimeout       =
#elastic.batchSize         =
```

Metrics Configuration

Make sure to set the proper `system.name` property in each `server.properties` file as well since this property will be used for the `appStack` tag. The system name will also be used as a prefix for fulltextsearch index name.


 Blanks will be replaced with `_`. Best practice: use 0-9A-Za-z.-

server.properties

```
#####
### System Settings                                     ###
#####
# Specifies the name of the system, e.g. Test System /Productive System / Demo / Poad
etc.
# The system name will also be used as a prefix for fulltextsearch index name.
# Blanks will be replaced with _. Best practice: use 0-9A-Za-z.-
system.name =
```

The Metrics configuration is possible in the **log4j2.xml** of the corresponding application server. Scroll to the "Metrics" section and replace the existing configuration with the settings below. Micrometer metrics can be enabled completely or for specific sub categories.

For example enabling **metrics.databaseTables.job** sub category and disabling the parent **metrics.databaseTables** metrics category, will disable all database table metrics except for database table job metrics.

 Use "ERROR" to disable and "INFO" to enable metrics.

log4j2.xml

1	<code><!-- ===== --></code>
2	<code><!-- Metrics --></code>
3	<code><!-- ===== --></code>
4	
5	<code><!-- Micrometer metrics can be enabled completely or for specific sub categories:</code>
6	<code>Use "ERROR" to disable and "INFO" to enable metrics.</code>
7	<code>- "metrics.cache" activates Cache metrics</code>
8	<code>- "metrics.communication" activates Communication Framework metrics</code>

```

9      - "metrics.databaseTables" activates database tables metrics:
10      - "metrics.databaseTables.masterCatalog"
11      - "metrics.databaseTables.supplierCatalogs"
12      - "metrics.databaseTables.structure"
13      - "metrics.databaseTables.otherEntities"
14      - "metrics.databaseTables.job"
15      - "metrics.databaseTables.meta"
16      - "metrics.dataGraph" activates DataGraph metrics
17      - "metrics.dataQuality" activates DataQuality metrics.
18      - "metrics.detailModel" activates DetailModel metrics
19      - "metrics.hibernate" activates Hibernate metrics
20      - "metrics.hibernate.query" activates Hibernate Query metrics
21      - "metrics.http" activates Http Request metrics.
22      - "metrics.http.response" activates Http Response metrics.
23      - "metrics.jetty" activates Jetty metrics
24      - "metrics.log4j2" activates Log4j2 metrics
25      - "metrics.login" activates Login metrics.
26      - "metrics.login.user" activates Login user metrics.
27      - "metrics.mediator" activates Mediator metrics
28      - "metrics.mediator.submediator" activates Sub-Mediator metrics
29      - "metrics.persistenceManager" activates PersistenceManager metrics
30      - "metrics.restClient" activates REST client metrics
31      - "metrics.auditTrail" activates Audit trail metrics
32      - "metrics.import" activates Import metrics
33      - "metrics.import.persistence" activates Import persistence
metrics
34      - "metrics.import.storage" activates Import storage metrics
35      - "metrics.triggerService" activates Trigger Service metrics
36      -->
37
38      <Logger name="metrics" level="INFO" additivity="false" />
39
40      <!-- Following metrics are disabled by default for performance reasons,
enable only for debugging -->
41      <!--Cache, Communication, http, hibernate -->
42      <Logger name="metrics.cache" level="ERROR" additivity="false" />
43      <Logger name="metrics.communication" level="ERROR" additivity="false" />
44      <Logger name="metrics.http" level="ERROR" additivity="false" />
45      <Logger name="metrics.hibernate" level="ERROR" additivity="false" />
46
47      <!-- Database Tables metrics are disabled by default, Except for Job, for
performance reasons, enable only for debugging-->
48      <Logger name="metrics.databaseTables" level="INFO" additivity="false" />
49      <Logger name="metrics.databaseTables.job" level="INFO" additivity="false"
/>
50      <Logger name="metrics.databaseTables.masterCatalog" level="ERROR"
additivity="false" />
51      <Logger name="metrics.databaseTables.supplierCatalogs" level="ERROR"
additivity="false" />
52      <Logger name="metrics.databaseTables.structure" level="ERROR" additivity="
false" />
53      <Logger name="metrics.databaseTables.otherEntities" level="ERROR"
additivity="false" />

```

```

54 <Logger name="metrics.databaseTables.meta" level="ERROR" additivity="false
55 " />
56 <!-- Sub mediator metrics are disabled by default for performance reasons,
57 enable only for debugging-->
57 <Logger name="metrics.mediator" level="INFO" additivity="false" />
58 <Logger name="metrics.mediator.submediator" level="ERROR" additivity="false" />

```

Import of Kibana Saved Objects

i Please note that from version Elastic Kibana 7.7 onwards, you might have to create a dedicated space for p360 metrics first before importing the saved objects. The imported settings, index pattern, visualizations and dashboards should reside in this space.

Once the above mentioned Product 360 server configurations are complete, the Kibana JSON objects for settings, index pattern, visualisations and dashboards can be imported.

For this open the **Kibana > Management > Saved Objects** page and Import the JSON objects "P360 Micrometer Kibana Dashboard Config" directory, in the following order:

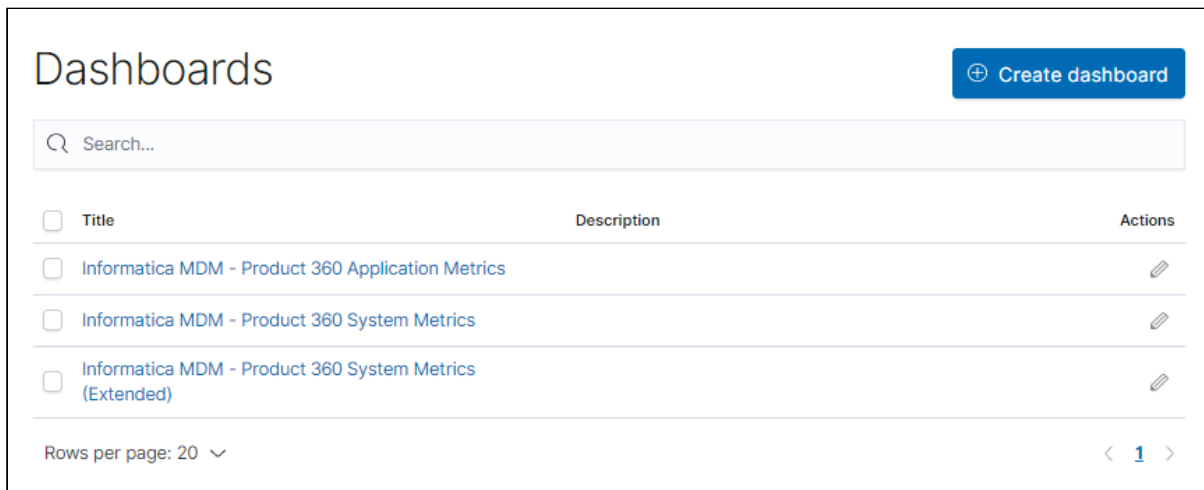
1. **p360_advanced_settings.ndjson**: Optimal settings for performant aggregation queries
2. **p360_kibana_index.ndjson**: The Kibana index pattern for "p360-metrics-idx" mapping the metrics files to the Product 360 metrics index
3. **p360_kibana_visualization.ndjson**: The individual Kibana metric visualisations
4. **p360_kibana_dashboard.ndjson**: The 3 Product 360 application and system monitoring dashboards

Once the above objects are successfully imported, ensure the following settings are enabled in the **Kibana > Management > Advanced Settings**

i Default index : p360-metrics-idx
 Index pattern placeholder : p360_metrics-
 Store URLs in session storage : On
 Maximum buckets : 20000

2.12.4.2 Dashboards

The Product 360 Kibana Dashboards should be available under **Kibana > Dashboards** as shown below :



The Product 360 Micrometer based monitoring is achieved through the following three Kibana Dashboards:


Dashboard name	Purpose
Informatica MDM - Product 360 Application Metrics	Product 360 specific application metrics that a Business administrator could use to monitor and track relevant statistics
Informatica MDM - Product 360 System Metrics	Product 360 server metrics that a System administrator could use to monitor and track relevant statistics
Informatica MDM - Product 360 System Metrics (Extended)	Low level Product 360 server metrics that a System administrator could use to monitor and track relevant statistics

Global Filters

In order to use the dashboards efficiently the following global filters are provided that will filter all metrics against the selection(s) of the filter.

Application Stack

The application stack filter allows the use of multiple P360 installations within the same Elasticsearch index. For example a single index can store the metrics for DEV, TEST, QA and PROD installations and the application stack filter allows easy switching between them.


 Please note that all metrics can be filtered against the application stack.

Metric Tag

appStack

Application Server

The application server filter allows the use of multiple P360 application servers within the same Elasticsearch index. All metrics will be filtered against the selection of this filter. For example a single index can store the metrics for a multi-server installation of P360, consisting of pim-server1 and pim-server2, and the application server filter allows easy switching between them.


 Please note that all metrics can be filtered against the application stack.

Metric Tag

appId

Entity Type

The entity type filter allows filtering (drill-down) of metrics regarding their corresponding repository entity types. For example it is possible to drill-down into all "item" (ArticleType) metrics.


 Please note that not all metrics are related to repository entity types and won't show any data while the entity type filter has been selected.

Metric Tag

entityType

Entity

The entity filter allows filtering (drill-down) of metrics regarding their corresponding repository entities. For example it is possible to drill-down into all "item" (Article) metrics.


 Please note that not all metrics are related to repository entities and won't show any data while the entity filter has been selected.

Metric Tag

entity

Database

The database filter allows filtering (drill-down) of metrics regarding any kind of database activity. For example it is possible to show all active connections for a specific database.


 Please note that not all metrics are related to the database and won't show any data while the database has been selected.

Metric Tag

pool

Message Type

The message type filters allows filtering (drill-down) of the communication framework for specific messages (every message in the communication framework has a specific message type). For example it is possible to show the amount and duration of rich client which request data from the database.

 Please note that only the communication framework metrics are message type specific, all other metrics won't show any data while the message type has been selected.

Metric Tag

messageType

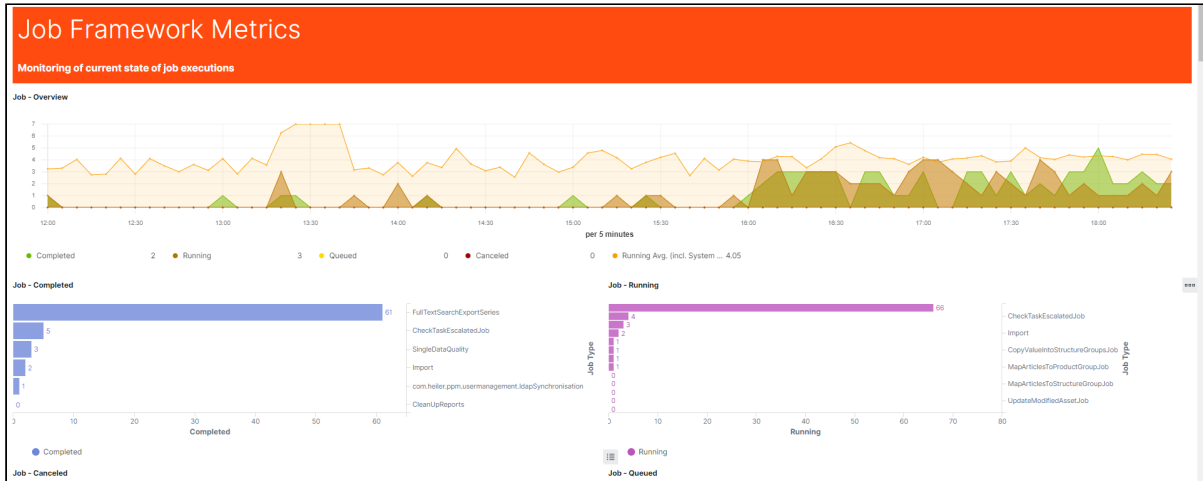
Application Metrics Dashboard

- [Job Framework Metrics](#) (see page 90)
- [Data Quality Metrics](#) (see page 90)
- [Service API Metrics](#) (see page 91)
- [Persistence Metrics](#) (see page 92)
 - [Datagraph](#) (see page 92)
 - [Detail Model Metrics](#) (see page 93)
 - [Persistence Manager](#) (see page 94)
 - [Mediator / Sub-Mediator](#) (see page 95)
- [Database Tables Statistics](#) (see page 97)
- [Message Queue Statistics](#) (see page 98)
- [Audit trail Metrics](#) (see page 99)

- [Trigger Service Metrics](#) (see page 99)
- [Import Metrics](#) (see page 100)

Job Framework Metrics

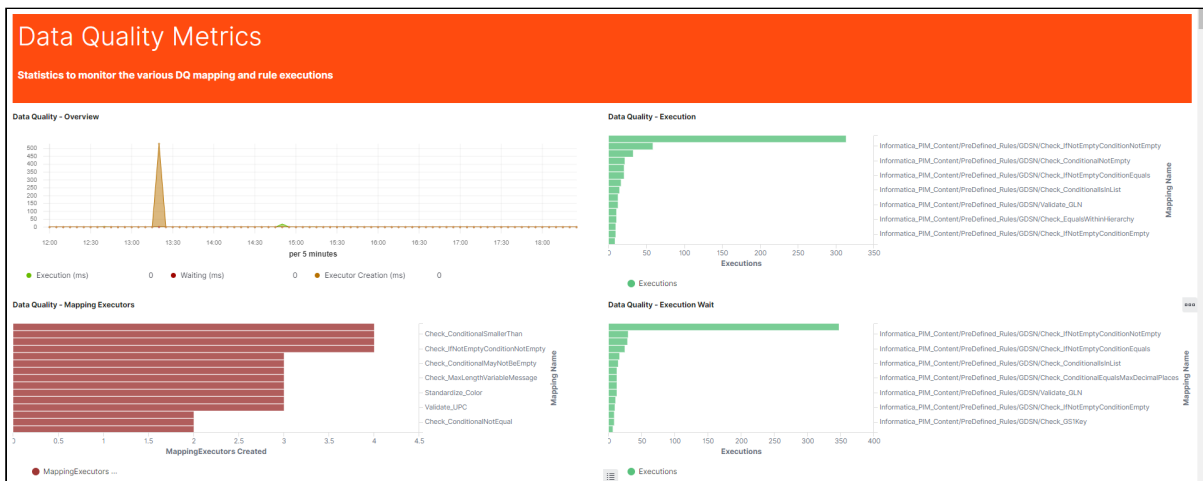
Monitoring of current state of job executions.



Metric Name	Metric Key	Description
Canceled	job_canceled	The amount of canceled jobs
Completed	job_completed	The amount of completed jobs
Queued	job_queued	The amount of queued jobs
Running	job_running	The amount of running jobs

Data Quality Metrics

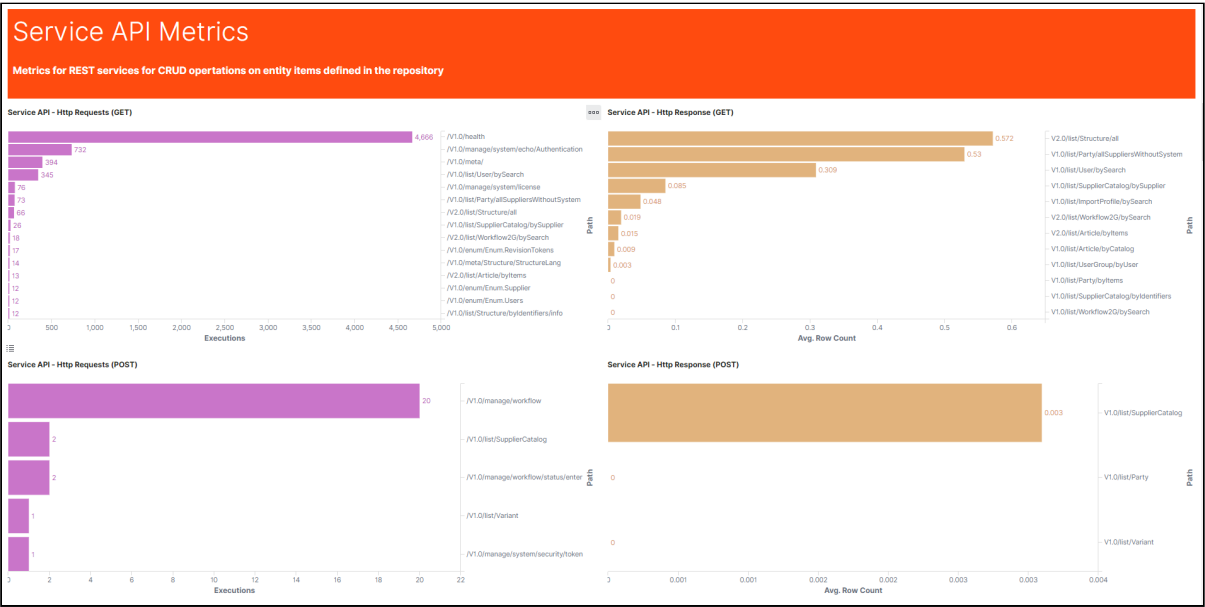
Statistics to monitor the various DQ mapping and rule executions.



Metric Name	Metric Key	Description
Mapping Execution	dataQuality_executing	The amount of Data Quality rules currently executing by the system
Mapping Executor Creation	dataQuality_executor_creation	The amount of Data Quality Mapping Executors created by the system
Mapping Execution Wait	dataQuality_waiting	The amount of time the Mapping Executors had to wait for a free execution slot

Service API Metrics

Metrics for REST services for CRUD operations on entity items defined in the repository.

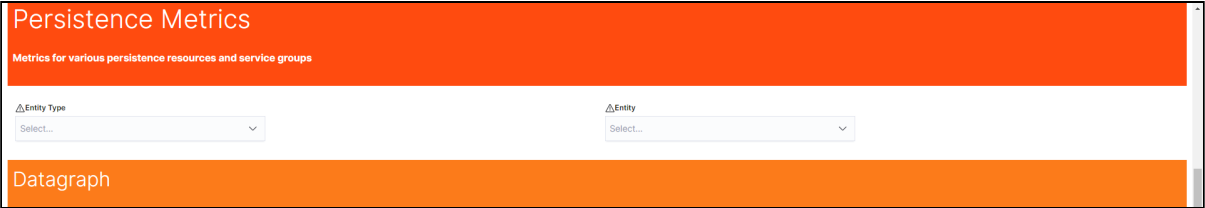


Metric Name	Metric Key	Description
HTTP Requests	httpRequest	The amount of HTTP requests
HTTP Responses	httpResponse_list	The amount of objects in the HTTP response

Persistence Metrics

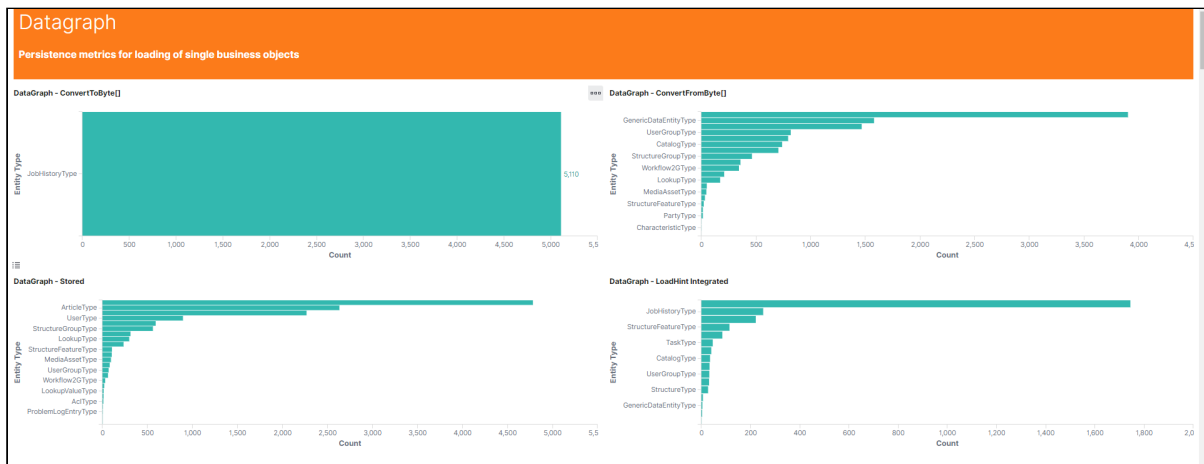
Metrics for various persistence resources and service groups.

The global persistence filters include both **EntityType** and **Entity** from the repository.



Datagraph

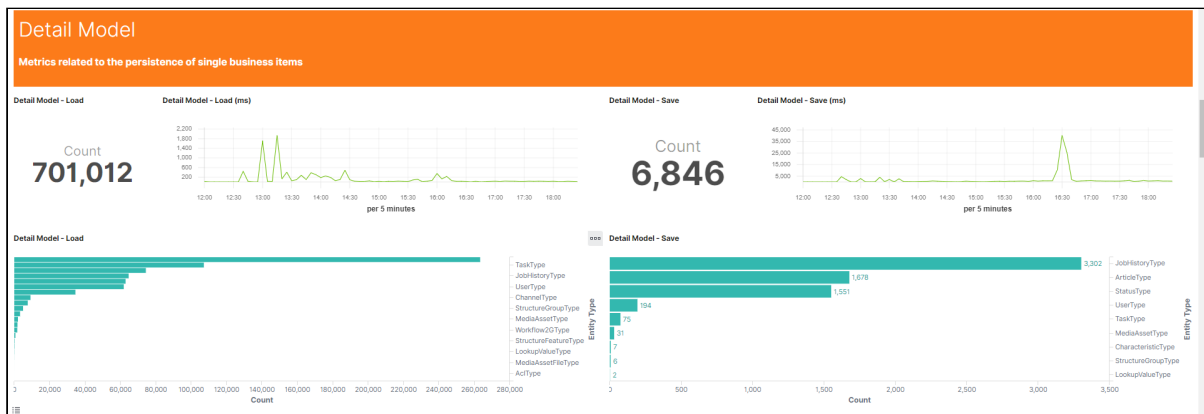
Persistence metrics for loading of single business objects.



Metric Name	Metric Key	Description
DataGraph - ConvertToByte[]	dataGraph_convertFromByteArray	The amount of datagraph models (i.e. a single business object) deserialised
DataGraph - ConvertFromByte[]	dataGraph_convertToByteArray	The amount of datagraph models (i.e. a single business object) serialised
DataGraph - LoadHint	dataGraph_loadHintIntegrated	The amount of datagraph models (i.e. a single business object) read from the database
DataGraph - Stored	dataGraph_stored	The amount of datagraph models (i.e. a single business object) stored to the database

Detail Model Metrics

The Detail Model is the business object representation of a single object (e.g. a single item). A high load count is an indication that the application is loading a lot of single objects, while a high save count is an indication that the application is modifying a lot of single objects.

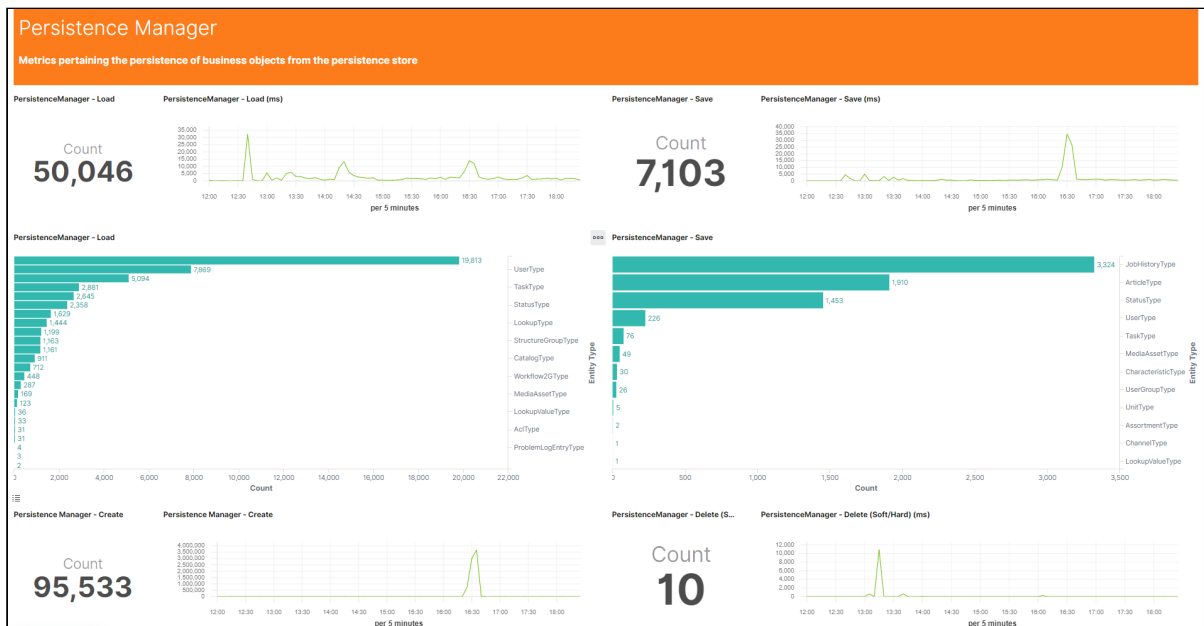


Metric Name	Metric Key	Description
Detail Model Load	detailModel_load	The amount of detail models (i.e. a full business entity) read from the database
Detail Model Save	detailModel_save	The amount of detail models (i.e. a full business entity) stored to the database

Persistence Manager

The PersistenceManager is the instance responsible for retrieving the business objects from the persistence store.

For example creating a new item will show up as **PersistenceManager - Create**, while loading an existing item will show up as **PersistenceManager - Load**.



Metric Name	Metric Key	Description
PersistenceManager Create	persistenceManager_create	The amount of top level business entities created by the system
PersistenceManager Delete	persistenceManager_delete	The amount of top level business entities deleted by the system
PersistenceManager Read	persistenceManager_load	The amount of top level business entities loaded by the system
PersistenceManager Write	persistenceManager_save	The amount of top level business entities saved by the system

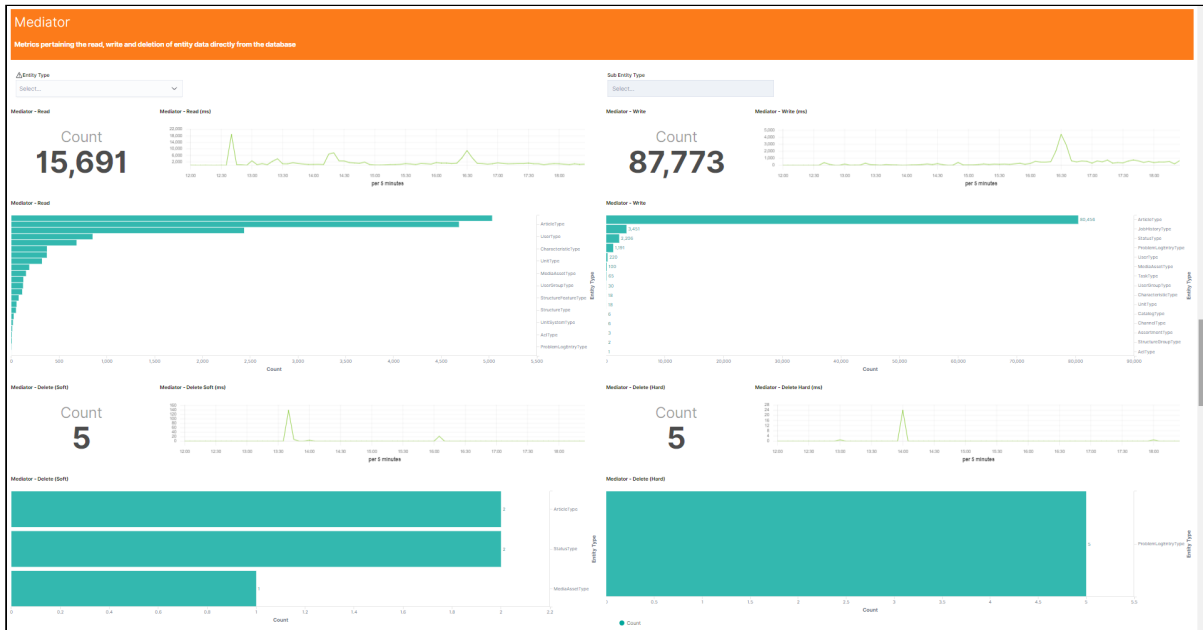
Mediator / Sub-Mediator

Mediators / Sub-Mediators are responsible for directly reading, writing and deleting Entity/ Sub-Entity data from the database.

For example creating a new item will show up as **Mediator/ SubMediator - Write**, while loading an existing item will show up as **Mediator/ SubMediator - Read**.



Sub-Mediator metrics are disabled by default to prevent excessive writes to the elastic search index. If required for debugging purposes, these metrics can be enabled by setting the corresponding log4j.xml entry to "INFO"



Metric Name	Metric Key	Description
Mediator Delete Hard	mediator_delete_hard	The amount of root entity hard delete operations
Mediator Delete Soft	mediator_delete_soft	The amount of root entity soft delete operations
Mediator Read	mediator_read	The amount of root entity load operations
Mediator Write	mediator_write	The amount of root entity save operations
SubMediator Delete Hard	submediator_delete_hard	The amount of sub entity hard (physical) delete operations

Metric Name	Metric Key	Description
SubMediator Delete Soft	submediator_delete_soft	The amount of sub entity soft delete operations
SubMediator Read	submediator_read	The amount of sub entity load operations
SubMediator Write	submediator_write	The amount of sub entity save operations

Database Tables Statistics

The actual amount of rows of all configured database tables are available as metrics.

Since counting the rows of all tables in real time would be a very expensive operation, the row count is actually being read directly from the database statistics itself. The query which is being used to fetch the row count from the database statistics will be executed in a five minute interval, i.e. counters will be updated every five minutes. While Microsoft SQL Server usually has auto-update statistics, the statistics on Oracle databases usually must be refreshed periodically.

Database Tables Statistics					
The actual amount of rows of all configured database tables					
JobHistory 679	ProblemLogEntry 8,546	Workflow2G 14	ProcessStatusEntry 116	Report 413	
Status 1,154,239	StatusEntry 19,874,377	Workflow2GStatus 30	Task 92	PublicationStatus 0	

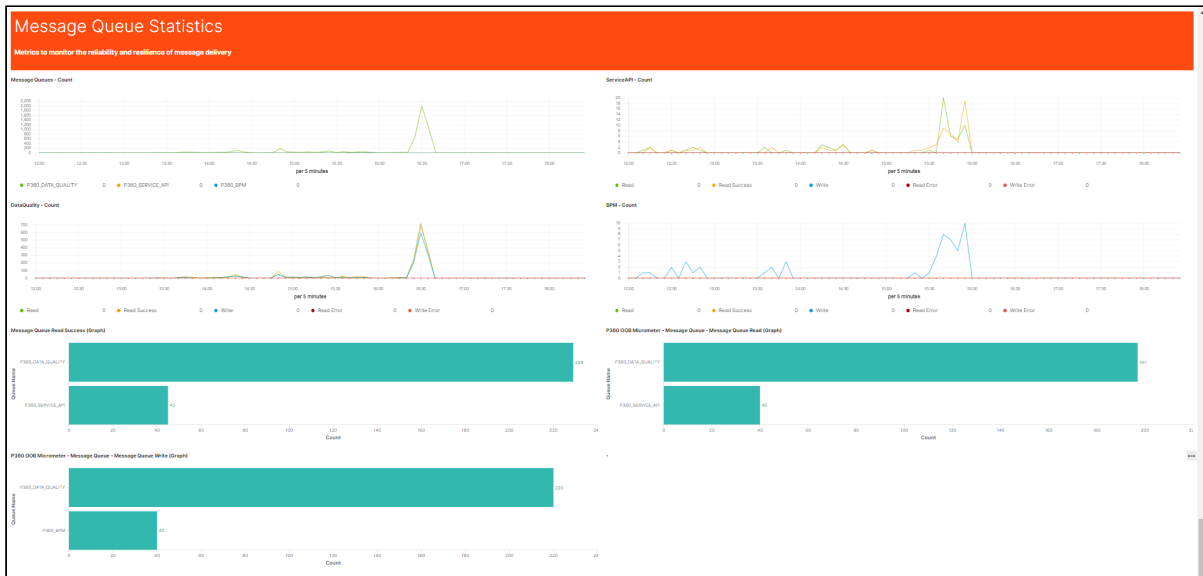
Metric Name	Metric Key	Description
Database Table Size	db_table_size	The amount of rows currently residing in the specified database table



Please note that for performance reasons this metric is being queried from the database statistics, so the real value could be different.

Message Queue Statistics

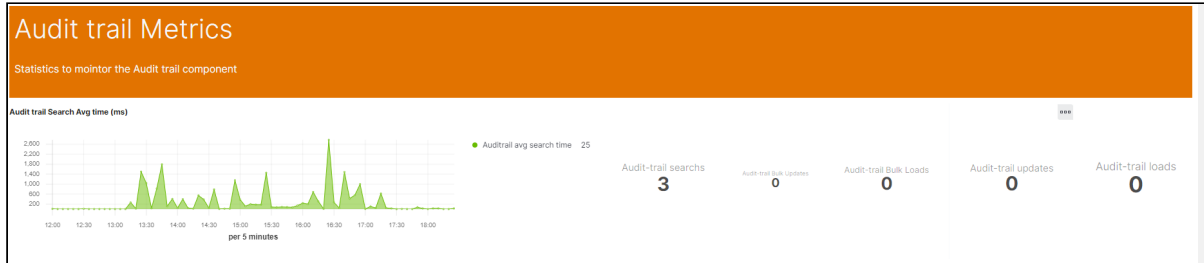
Metrics to monitor the reliability and resilience of message delivery in Service API, Data Quality and BPM message queues.



Metric Name	Metric Key	Description
Messages Read	queue_messages_read	The amount of messages read from the message queue
Messages Read Successfully	queue_messages_read_success	The amount of messages successfully read from the message queue
Messages Read Error	queue_messages_read_error	The amount of messages read with error from the message queue
Messages Write	queue_messages_write	The amount of messages written to the message queue
Messages Write Error	queue_messages_write_error	The amount of messages failed to write to the message queue

Audit trail Metrics

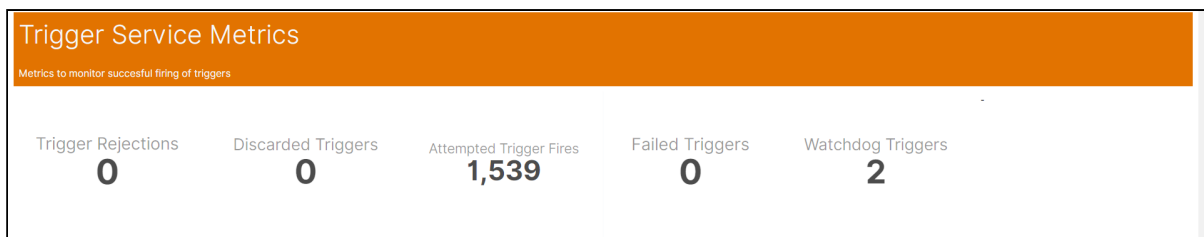
Statistics to monitor the Audit trail component.



Metric Name	Metric Key	Description
Audit trail search	auditTrail_searchService_search	timer measuring the average time taken for audit trail search
Audit trail create (Bulk)	auditTrailIndexService_bulkCreate	timer measuring bulk creations in the audit trail index
Audit trail update (Bulk)	auditTrailIndexService_bulkUpdate	timer measuring bulk updates to the audit trail index
Audit trail create	auditTrailIndexService_create	timer measuring individual creations in the audit trail index
Audit trail update	auditTrailIndexService_update	timer measuring updates to the audit trail index

Trigger Service Metrics

Metrics to monitor successful firing off triggers.



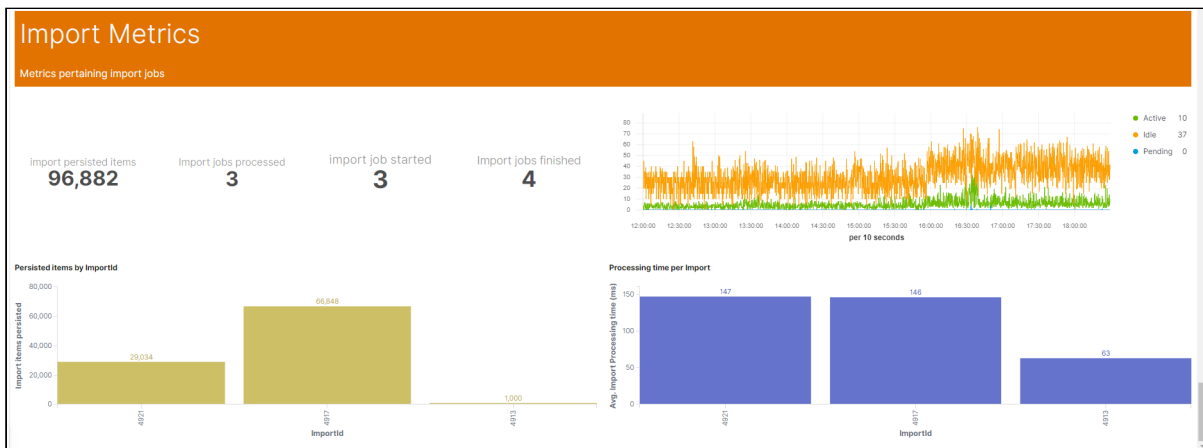
Metric Name	Metric Key	Description
Trigger Rejections	triggerService.rejectedAsExecutorServiceFull	It is the counter for number of requests that are directed to Trigger Rejection policy as Trigger executor service is under high load.
Discarded Triggers	triggerService.discardedAsNonJobThread	It is the counter for number of trigger execution requests that are discarded as trigger service is under high load and this request is initiated from UI/OTHER initiator module. So these are the triggers that will be fired from trigger watchdog.
Watchdog Triggers	triggerService_pickedByTriggerWatchdog	Counter for number of triggers that are picked for firing by trigger watchdog
attemptedToFire	triggerService.attemptedToFire	Counter for number of triggers firing attempts made by Trigger Service
triggerFireFailed	triggerService.triggerFireFailed	Counter for number of trigger firing attempts that failed

Import Metrics

Metrics pertaining import jobs.



Please note import Statistics are only available from Product 360 10.1 HF1 onwards



Metric Name	Metric Key	Description
Import - Items Persisted	importer_persistence_item	The amount of items persisted to the database during the import
Import - Storage Get	importer_storage_get	The amount of objects retrieved from the import persistence model
Import - Storage Put	importer_storage_put	The amount of objects stored to the import persistence model
Import - Job Finished	importer_job_finished	The amount of finished import jobs
Import - Job Processing	importer_job_processed	The amount of currently active import jobs

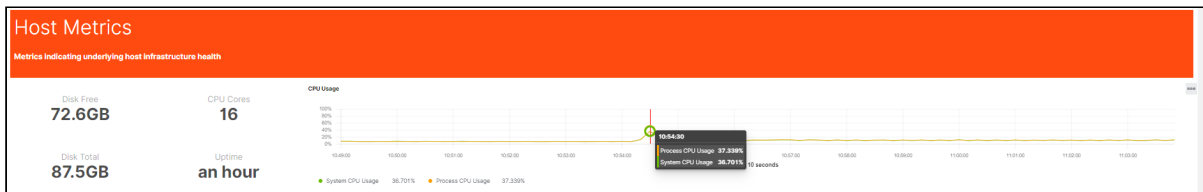
System Metrics Dashboard

- [Host Metrics](#) (see page 102)
- [JVM Statistics - Memory](#) (see page 102)
- [JVM Garbage Collection Statistics](#) (see page 103)
- [Login Session Statistics](#) (see page 104)
- [Jetty \(Web Service Framework\) Statistics](#) (see page 105)
- [Connection Pool Statistics](#) (see page 106)
 - [Global filter : Database](#) (see page 106)
 - [Database HikariCP Statistics](#) (see page 107)
 - [Database FlexyPool Statistics](#) (see page 108)
- [Thread Pool Statistics](#) (see page 108)

- [Communication Framework](#) (see page 109)
- [Global Filter : Message Type](#) (see page 109)
- [Fragment Framework](#) (see page 110)
- [Reporting Framework](#) (see page 110)
- [Trigger Framework](#) (see page 111)
- [Audit Trail Executer](#) (see page 111)

Host Metrics

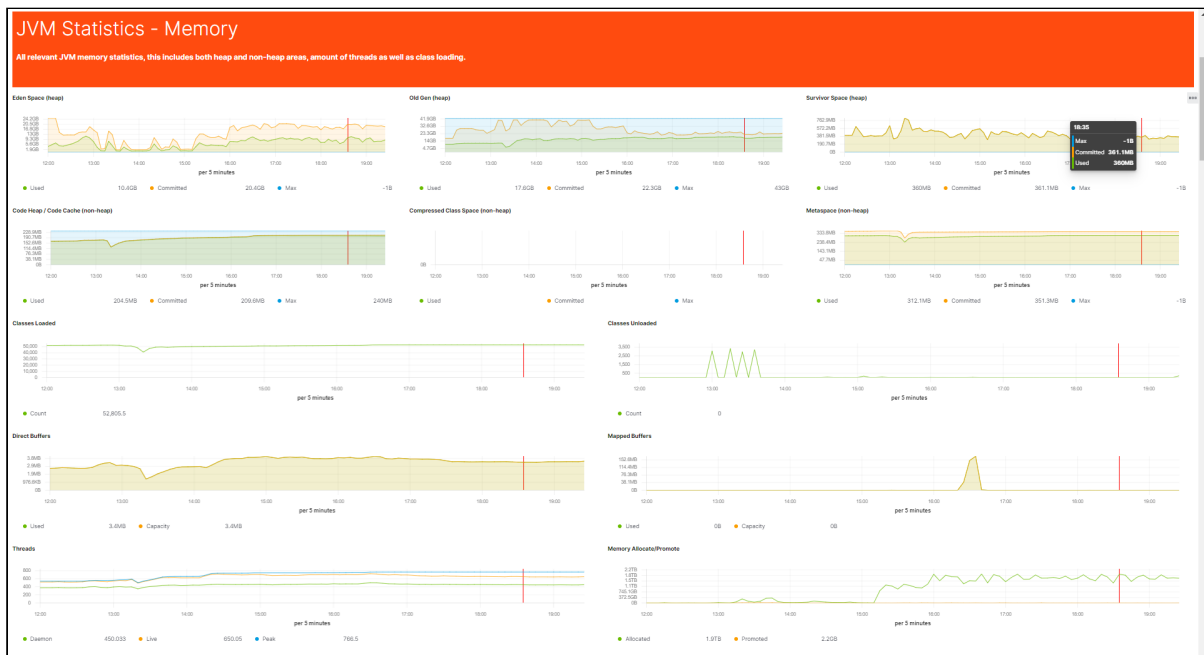
Metrics indicating underlying host infrastructure health.



Metric Name	Metric Key	Description
Disk Space Free	disk_free	The available disk space of the drive where Product 360 is installed
Disk Space Total	disk_total	The total disk space of the drive where Product 360 is installed
CPU Count	system_cpu_count	The available system processors
CPU Usage	system_cpu_usage	The current system CPU utilization
Process Uptime	process_uptime	The amount of time the Product 360 service is running

JVM Statistics - Memory

All relevant JVM memory statistics, this includes both heap and non-heap areas, amount of threads as well as class loading.

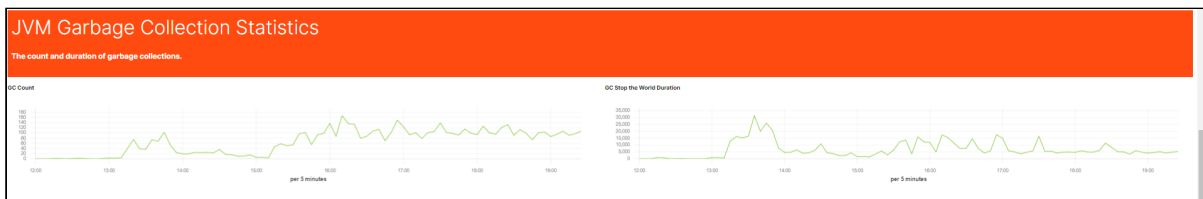


Metric Name	Metric Key	Description
Classes Loaded	jvm_classes_loaded	The amount of classes loaded by the JVM.
Classes Unloaded	jvm_classes_unloaded	The amount of classes unloaded by the JVM.
Buffer Memory	jvm_buffer_memory_used	The amount of buffer memory used by the JVM.
Memory Usage	jvm_memory_used	The amount of memory used by the JVM.
Threads	jvm_threads_daemon	The amount of threads used by the JVM.

JVM Garbage Collection Statistics

JVM garbage collection metrics include the count and duration of garbage collections.

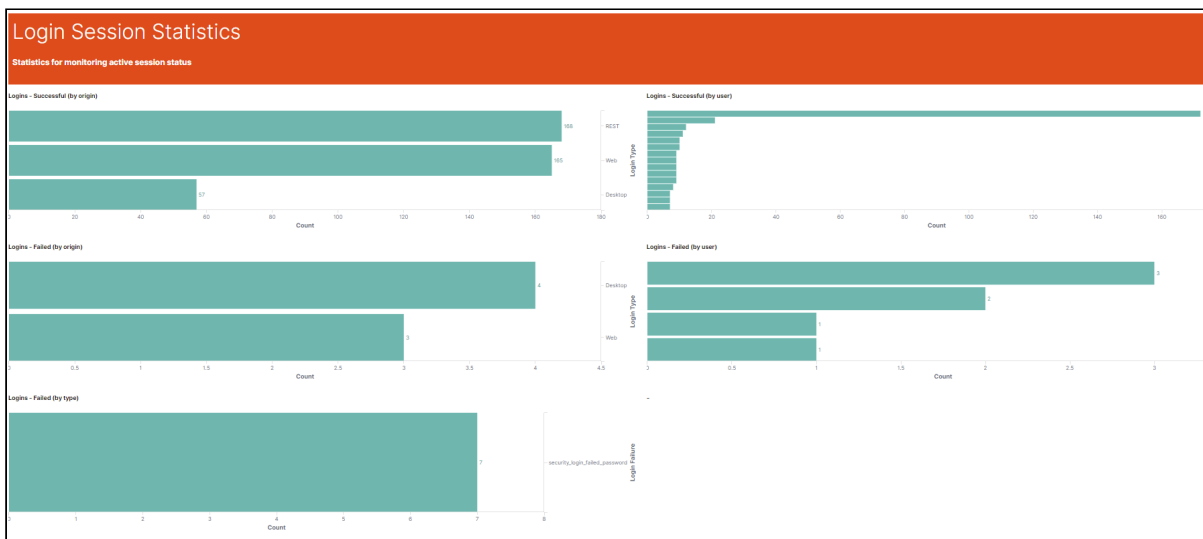
While a high count can be fine, a high duration usually suggests that the application is running low on heap memory.



Metric Name	Metric Key	Description
GC Count	jvm_gc_pause	The amount of garbage collections executed by the JVM
GC Stop the World Duration	jvm_gc_pause	The duration of the garbage collections executed by the JVM

Login Session Statistics

Statistics for monitoring active session status.

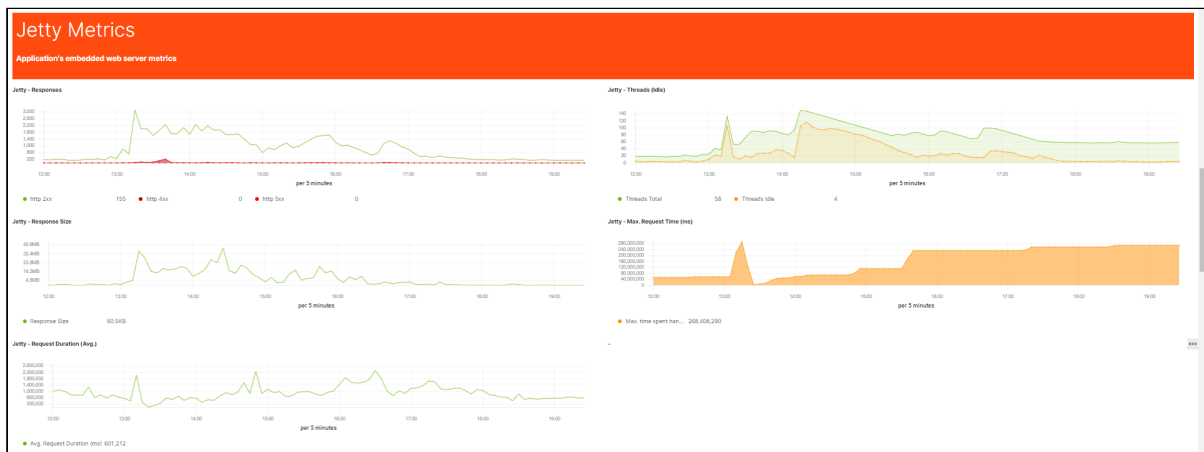


Metric Name	Metric Key	Description
Login Successful	security_login_successful	The amount of successful logins

Metric Name	Metric Key	Description
Login Failed	security_login_failed_admin	The amount of failed logins for Administrator account
	security_login_failed_inactive	The amount of failed logins for inactive accounts
	security_login_failed_password	The amount of failed logins due to wrong password
	security_login_failed_permission	The amount of failed logins due to insufficient permissions
	security_login_failed_other	The amount of failed logins for any other reason

Jetty (Web Service Framework) Statistics

Application's embedded web server metrics.



Metric Name	Metric Key	Description
Request Duration	jetty_requests	The average duration of a web service request

Metric Name	Metric Key	Description
Request Max. Time	jetty_dispatched_time_max	The maximum duration of a web service request
Response Size	jetty_responses_size	The size of response payload of the web service request
Responses	jetty_responses	The amount of responses to web service requests
Threads	jetty_threads_current	The current amount of JVM threads

Connection Pool Statistics


Metrics pertaining connection events such as creation, usage, return to connection pool etc.

Every data source has its own connection pool. A connection must be created, used, and returned to the connection pool and there are metrics for all events.

Important connection pool metrics are the amount of open connections, and how many of these connections are active or idling. It is important to watch for anything out of the ordinary, i.e. connection timeouts, many active connections and especially if any overflow of the connection pool takes place.

Global filter : Database

The database filter allows filtering (drill-down) of metrics regarding any kind of database activity. For example it is possible to show all active connections for a specific database.

 Please note that not all metrics are related to the database and won't show any data while the database has been selected.

Metric Tag

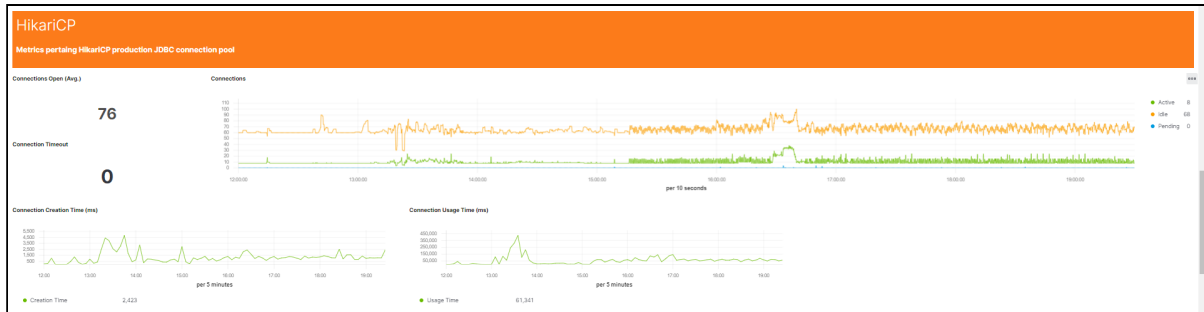
pool

Connection Pool Statistics
Metrics pertaining connection events such as creation, usage, return to connection pool etc

 Database

Database HikariCP Statistics

Metrics pertaining HikariCP production JDBC connection pool.



Metric Name	Metric Key	Description
Connection Acquire Time	hikaricp_connections_acquire	The amount of time it took to acquire a connection from the connection pool
Connection Creation Time	hikaricp_connections_creation	The amount of time it took to create a connection for the connection pool
Connection Timeout Count	hikaricp_connections_timeout	The amount of timeouts while requesting a connection from the connection pool
Connection Usage Time	hikaricp_connections_usage	The amount of time the connection from the connection pool was in use
Connections	hikaricp_connections_active	The amount of active connections in the connection pool
Connections Size	hikaricp_connections	The amount of connections in the connection pool

Database FlexyPool Statistics

Monitoring and failover metrics provide by Flexypool data source proxy.



Metric Name	Metric Key	Description
Concurrent Connection Request	concurrentConnectionRequestsHistogram	The amount of concurrent connections requested by the connection pool
Concurrent Connections	concurrentConnectionsHistogram	The amount of concurrent connections used by the connection pool
Max Pool Size	maxPoolSizeHistogram	The maximum connection pool size
Overflow Pool Size	overflowPoolSizeHistogram	The overflow connection pool size (i.e. how much the maximum pool size has been exceeded)
Retry Attempts	retryAttemptsHistogram	The amount of retries required in order to request a connection from the connection pool

Thread Pool Statistics

Metrics for monitoring concurrent execution performance

Thread Pool Statistics

Metrics for monitoring concurrent execution performance

Metric Name	Metric Key	Description
Executor Active	executor_active	The amount of active worker threads in the thread pool
Executor Completed	executor_completed	The amount of completed worker tasks of the thread pool
Executor Queued	executor_queued	The amount of queued worker tasks of the thread pool
Executor Pool Size	executor_pool_size	The thread pool size

Communication Framework


The Communication Framework is responsible for sending data between application servers and application server and rich clients.

Every communication message shows up in this thread pool, and it is important to watch out for queuing of any kind.

Global Filter : Message Type

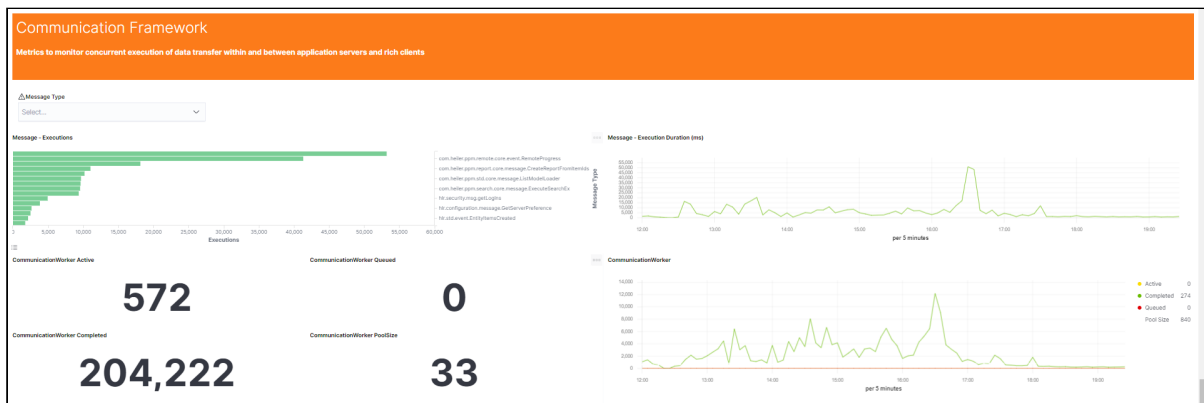
The message type filters allows filtering (drill-down) of the communication framework for specific messages (every message in the communication framework has a specific message type).

For example it is possible to show the amount and duration of rich client which request data from the database.

 Please note that only the communication framework metrics are message type specific, all other metrics won't show any data while the message type has been selected.

Metric Tag

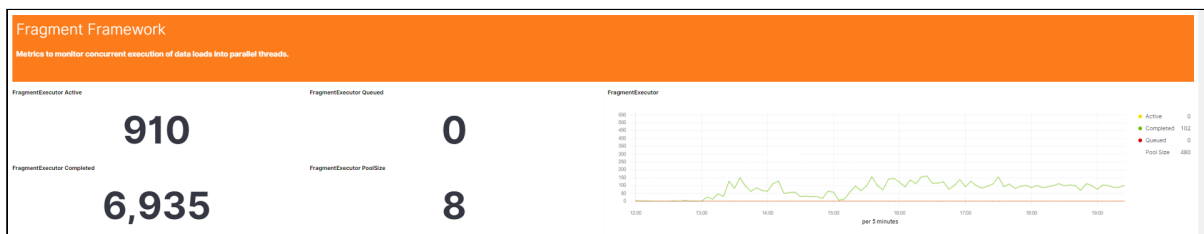
messageType



Metric Name	Metric Key	Description
Message Execution	communication_message	The amount of messages executed by the system

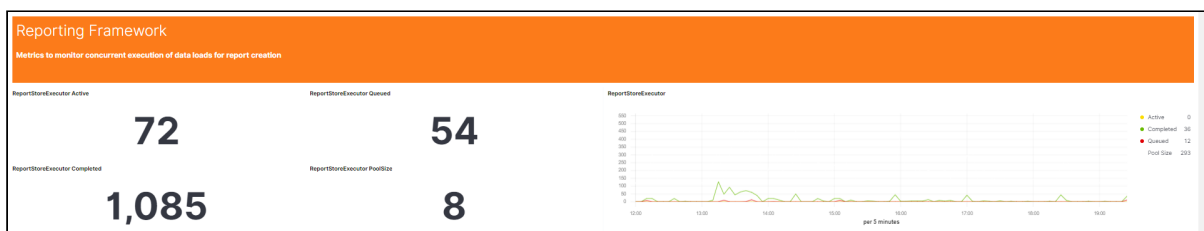
Fragment Framework

The Fragment Manager has a thread pool which supports loading data in parallel threads. Every parallel loading request of will show up in this thread pool, and it is important to watch out for huge queues.



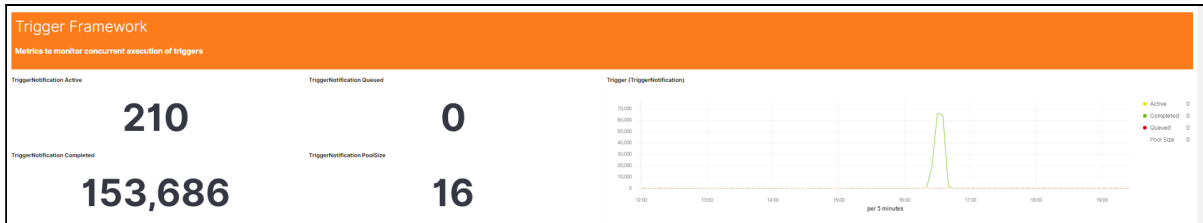
Reporting Framework

The Reporting Framework needs to create reports (i.e. list of object ids) in order to load any kind of data. When these reports exceed a specific size they will be created in parallel threads, and it is important to watch out for huge queues.



Trigger Framework

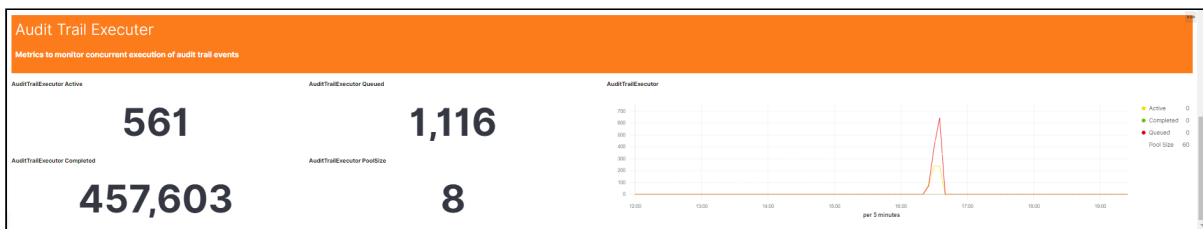
Metrics to monitor concurrent execution of triggers.



Audit Trail Executer

Every create, change or delete event in the application causes creation of an audit trail summary which will be consumed by triggers, workflows and audit trail.

Every audit trail event will show up in this thread pool, and it is important to watch out for huge queues.



System Metrics Dashboard - Extended

- [Cache Statistics](#) (see page 111)
 - [EntityProxyCache](#) (see page 112)
 - [Persistence Manager Cache](#) (see page 112)
 - [DataGraphContainerCache](#) (see page 113)
 - [Report Cache](#) (see page 113)
- [Hibernate Statistics](#) (see page 114)
- [Log4j2 Logging Statistics](#) (see page 116)

Cache Statistics

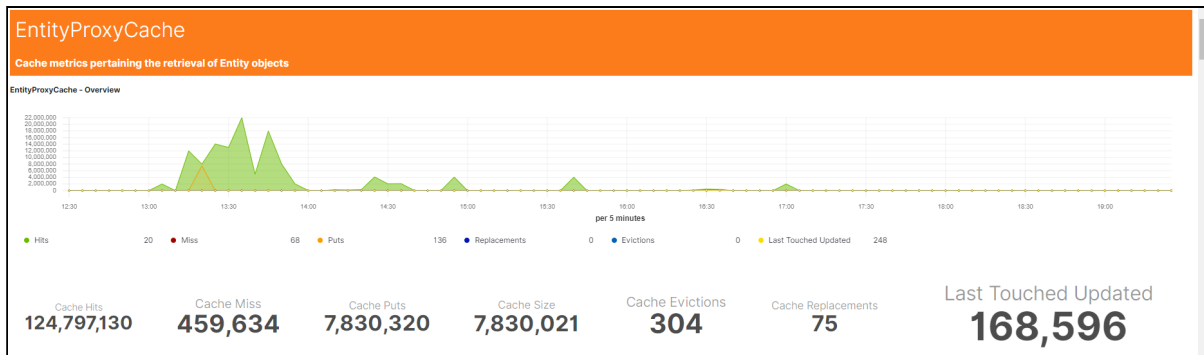
Metrics for various cache resources related to loading business objects. Cache objects expire upon modification or in case they have aged out. A high Cache miss rate should be investigated as well as Cache loads failure.



Metric Name	Metric Key	Description
Cache Evictions	cache_evictions	The amount of elements evicted by the cache
Cache Gets	cache_gets	The amount of elements requested by the cache
Cache Loads Duration	cache_load_duration	The amount of time it took the cache to load the element
Cache Loads	cache_load	The amount of elements loaded by the cache
Cache Puts	cache_puts	The amount of elements added to the cache
Cache Size	cache_size	The total amount of elements in the cache

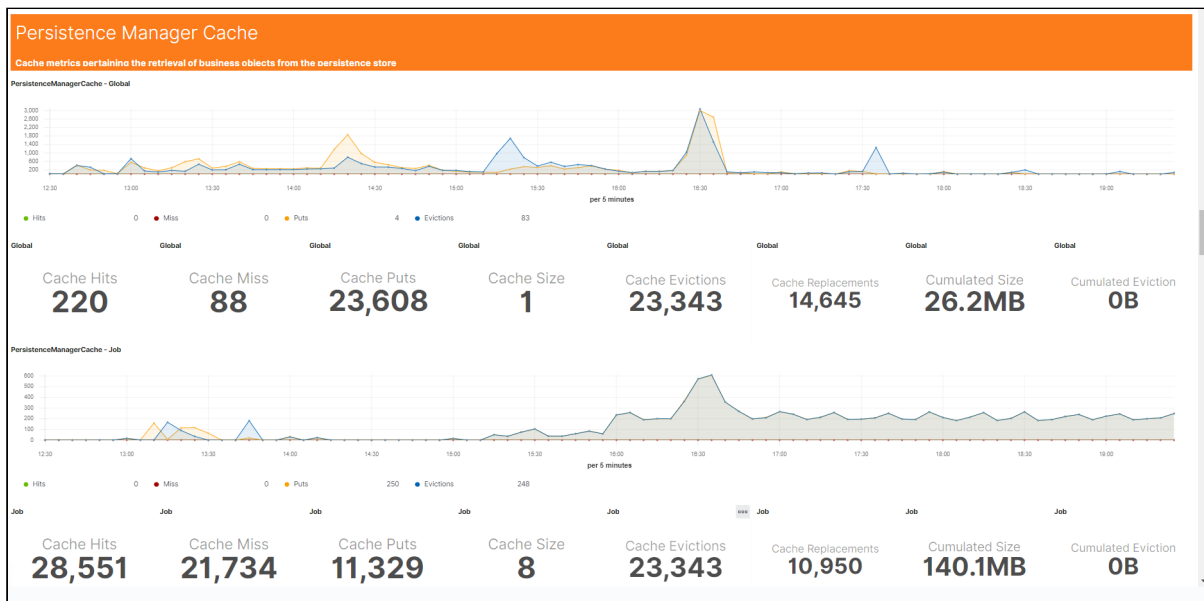
EntityProxyCache

Cache metrics pertaining the retrieval of Entity objects.



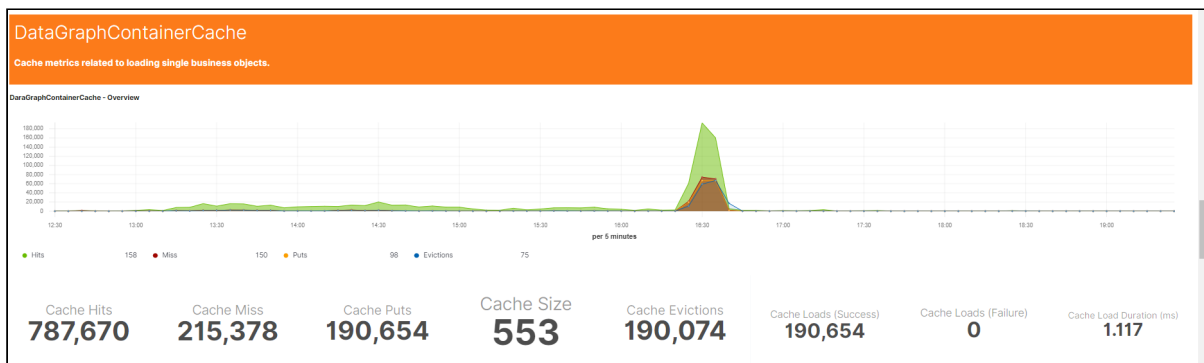
Persistence Manager Cache

Cache metrics pertaining the retrieval of business objects from the persistence store.



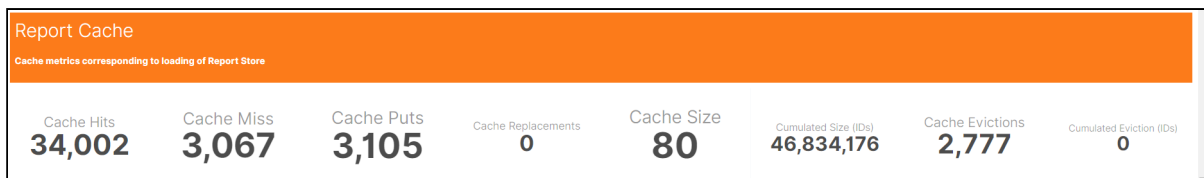
DataGraphContainerCache

Cache metrics related to loading single business objects.



Report Cache

Cache metrics corresponding to loading of Report Store.

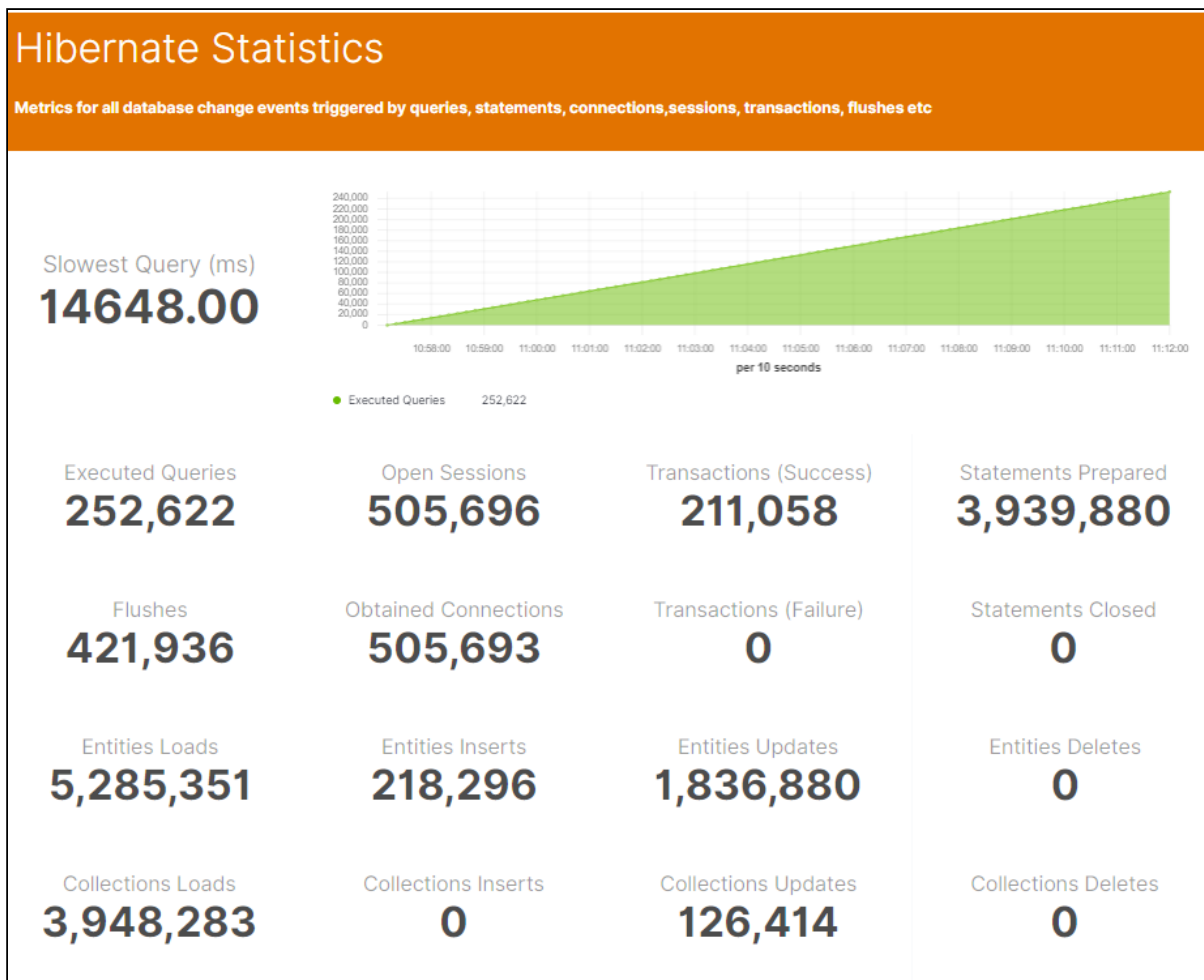


Hibernate Statistics

Metrics for all database change events triggered by queries, statements, connections, sessions, transactions, flushes etc.

Hibernate is the persistence service of the application and responsible for loading, saving and deleting data from the databases. Any change to the database requires a query or a statement, a connection, a session, a transaction, a flush, and there are metrics for all events.

Important Hibernate metrics are the amount of queries executed and especially queries which take a lot of time.



Metric Name	Metric Key	Description
Collections Deletes	hibernate_collections_deletes	The amount of collections (i.e. records from 1:n relationships) deleted by Hibernate

Metric Name	Metric Key	Description
Collections Inserts	hibernate_collections_inserts	The amount of collections (i.e. records from 1:n relationships) created by Hibernate
Collections Loads	hibernate_collections_loads	The amount of collections (i.e. records from 1:n relationships) loaded by Hibernate
Collections Updates	hibernate_collections_updates	The amount of collections (i.e. records from 1:n relationships) updated by Hibernate
Connections Obtained	hibernate_connections_obtained	The amount of database connections obtained by Hibernate
Entities Deletes	hibernate_entities_deletes	The amount of entities (i.e. top level business entities) deleted by Hibernate
Entities Inserts	hibernate_entities_inserts	The amount of collections (i.e. top level business entities) created by Hibernate
Entities Loads	hibernate_entities_loads	The amount of collections (i.e. top level business entities) loaded by Hibernate
Entities Updates	hibernate_entities_updates	The amount of collections (i.e. top level business entities) updated by Hibernate
Flushes	hibernate_flushes	The amount of flushes (i.e. synchronizing the database with the current state) executed by Hibernate
Query Executions	hibernate_query_executions	<p>The amount of single database queries executed by Hibernate</p> <p><i>Please note that enabling Hibernate Query Metrics results in noticable performance penalties</i></p>

Metric Name	Metric Key	Description
Query Max. Execution Time	hibernate_query_executions_max	The maximum duration of each single database query executed by Hibernate <i>Please note that enabling Hibernate Query Metrics results in noticable performance penalties</i>
Query Max. Rows	hibernate_query_executions_rows	The maximum amount of rows retrieved of each single database query executed by Hibernate <i>Please note that enabling Hibernate Query Metrics results in noticable performance penalties</i>
Session Open	hibernate_sessions_open	The amount of sessions opened (i.e. acquiring a connection to the database) by Hibernate
Statements Closed/ Prepared	hibernate_statements	The amount of statements prepared and closed by Hibernate
Transactions	hibernate_transactions	The amount of transactions executed by Hibernate

Log4j2 Logging Statistics

Metrics tracking the creation of all log event. The creation of any log event (TRACE to ERROR) will be shown here.



Metric Name	Metric Key	Description
DEBUG Logs	log4j2_events (level:debug)	The amount of DEBUG log entries written
ERRORS Logs	log4j2_events (level:error)	The amount of ERROR log entries written
INFO Logs	log4j2_events (level:info)	The amount of INFO log entries written
TRACE Logs	log4j2_events (level:trace)	The amount of TRACE log entries written
WARN Logs	log4j2_events (level:warn)	The amount of WARNING log entries written

2.12.4.3 Best Practices and Recommendations

For the optimal usage of Micrometer metrics and the Kibana dashboard, the following recommendations are made:

- Ensure that **elastic.step** in **micrometer.properties** server settings is not less than 1 min, as the defaults mentioned earlier. This setting controls the frequency with which metrics are stored to the elastic search index. Too short an interval could lead to increased elasticsearch index sizes
- The "**Informatica MDM - Product 360 System Metrics (Extended)**" gives access to low level monitoring of Caches, Hibernate etc. These metrics are disabled by default to prevent excessive elasticsearch index sizes and performance issues. These metrics should be enabled primarily for debugging purposes.
- When querying statistics for a time range of over multiple days, ensure that the "Maximum buckets" setting in **Kibana > Management > Advanced Settings** is set to a high number (greater than 20000 as suggested in the defaults)

2.13 Trigger Backpressure

2.13.1 Introduction

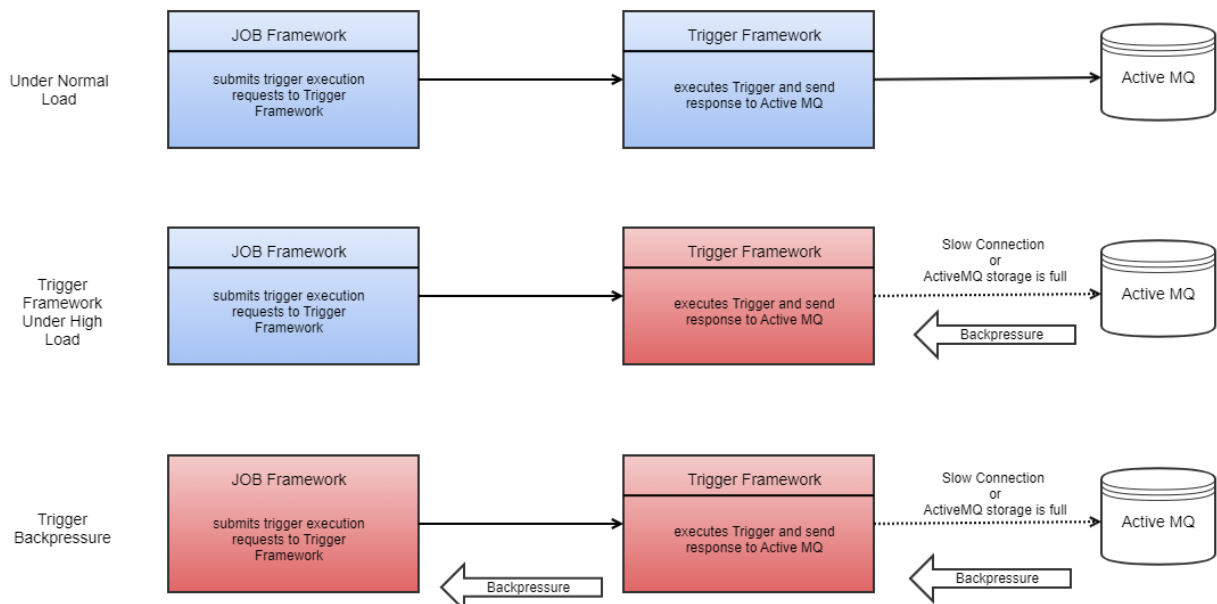
Trigger framework goes into high load situation when the rate at which it is receiving trigger requests is more than the rate at which it can execute them. When such a situation arises, Trigger Framework has the capability to propagate it's load to Job framework (which is producer of trigger execution requests) and make it little slow in bringing new requests. This propagation of high load from Trigger Framework to it's caller Job Framework is known as Trigger Backpressure.

2.13.2 When does Backpressure scenario happens?

Trigger framework can execute some requests parallelly(default value is 3 times of available number of CPU cores) and can queue some requests(default value is 10000) to be executed as soon as any of execution threads in trigger framework is available. When Trigger Framework is working on it's full potential and still not able to catch up with all the request that Job Framework put on it, then Trigger Backpressure kicks in.

i As soon as Trigger Backpressure kicks in, it marks the Trigger Framework under high load.

Let's see high level diagram depicting this behavior.



2.13.3 What happens during backpressure?

As seen in the above diagram, high load from Trigger Framework is propagated back to Job Framework. This means job framework is now aware that it has to slow down request submission to Trigger Framework. Not just that, Job Framework help trigger Framework by executing some of the triggers by itself(triggers whose initiator module is other than UI or OTHER), thus further reducing load on trigger framework. No other service of Product 360 is affected or slowed due to this. Once Trigger Framework comes out of high load state to normal state, Job Framework is also notified and it resumes it's normal operations.

Let's understand this with a scenario :

- Assume we started Import of 10k items and we have an Entity Created trigger configured which will fire after each item import. So we will have 10k trigger fire requests.
- Once import is started, connection between PIM server and Active MQ is lost.
- Trigger Framework will come under high load after some time as it has messages to write in Active MQ which is down.
- But Job Framework is still working on it's full potential, so injecting more requests in trigger framework.
- After some time, as we have trigger backpressure functionality, high load intimation will reach Job framework.
- Job Framework will slow down and eventually Import operation will be stalled.
- We will see trigger service under high load logs in console which signifies that backpressure has kicked in.
- All other functionality in PIM will still work normally.
- Once the lost connection is resumed, Trigger Framework will come out of high load.
- Eventually Job Framework will be notified and it will resume the Import operation.
- We will see that Import of 10K items are completed.



Trigger Framework comes out of high load state automatically after 5 mins from last time it has entered High Load state.

2.13.4 Log Messages during Trigger Backpressure

Different log messages are logged based on different types of trigger requests.

- **Request from Job Framework :** If Trigger backpressure kicks in and we have a new trigger request with Initiator module other than UI or OTHER (like IMPORT, MERGE, SERVICEAPI etc.), then we get following message in log

```
Trigger service underhigh load. Job thread will execute the trigger.
```

- **Request from UI or OTHER initiator module :** When the request is from UI or OTHER initiator module and Trigger Framework is under high load, then we get following log message

```
Trigger service underhigh load. Discarding the trigger, watchdog will execute it.
```

2.13.5 Trigger Watchdog

Trigger Watchdog is a thread which continuously runs throughout the lifecycle of server with sole motive of finding the discarded or un-executed trigger requests in the system and execute them.

- Trigger watchdog only executes these un-fired abandoned triggers when Trigger Service is not under high load.
- **Documents fetch batch size** : There is an upper limit on number of un-fired triggers watchdog fetches at a time for processing. Default size is : 1000.
- **Poll interval (in milliseconds) for trigger watchdog** : This is amount of time trigger watchdog waits before it again looks for un-fired triggers if it gets zero un-fired triggers in the current scan. Default value(in milliseconds) is : 300000
- **Max days for unfinished trigger processing** : This states the max number of days, beyond which older documents won't be fetched for processing of trigger if not already executed. Default value(in days) is : 7

Values of all 3 parameters defined above can be updated by either updating plugin_customization.ini file or at runtime using management beans.

Properties in plugin_customization.ini file :

```
# The value of this property determines for how many milli-seconds should the
watchdog wait
# until next execution in case it doesn't find any records.
# com.heiler.ppm.persistence.dr.server/
disasterRecovery.watchdog.trigger.pollIntervalInMills=300000

# This size determines how many documents watchdog should fetch per batch size while
retrieving.
# com.heiler.ppm.persistence.dr.server/disasterRecovery.watchdog.fetchBatchSize=1000

# The timestamp decides the maximum number of days beyond which older records won't
be picked up.
# com.heiler.ppm.persistence.dr.server/
disasterRecovery.watchdog.trigger.fromTimestampInDays=7
```

Updating values at runtime(does not require server restart through management beans using JConsole)

The screenshot shows the JConsole interface with the following details:

- Left Pane (MBeans):** A tree view showing the hierarchy of MBeans. The selected path is: `com.heiler.ppm.persistence.dr.server/disasterRecovery/watchdog/trigger/Documents fetch batch size`.
- Right Pane (Attribute value):** A table showing the current value of the selected attribute.

Name	Value
Documents fetch batch size	1000
- MBeanAttributeInfo:** A table showing the metadata for the attribute.

Name	Value
Attribute:	
Name	Documents fetch batch size
Description	This size determines how many documents will be fetched per batch while retrieving.
Readable	true
Writable	true
Is	false
Type	int

2.13.6 Trigger Framework Metrics

Trigger Framework captures some metrics while it is trying to fire triggers or enters in to high load situation. These metrics are mentioned in table below :

Metrics Name	Description
triggerService.rejectedAsExecutorServiceFull	counter for number of times Trigger Framework reaches high load situation.
triggerService.discardedAsNonJobThread	counter for number of trigger execution requests that are discarded as trigger service is under high load and this request is initiated from UI/OTHER initiator module. So these are the triggers that will be fired from trigger watchdog.
triggerService.pickedByTriggerWatchdog	Counter for total number of triggers that are picked for firing by trigger watchdog
triggerService.attemptedToFire	Counter for number of triggers firing attempts made by Trigger Service
triggerService.triggerFireFailed	Counter for number of trigger firing attempts that failed

3 Desktop Operation

Combines maintenance, tuning and troubleshooting for the PIM - Desktop

- [Maintenance Tasks](#) (see page 122)
- [Tuning](#) (see page 122)
 - [Performance optimization of the "Documents" view](#) (see page 122)
 - [Filter](#) (see page 123)
 - [Import](#) (see page 124)
- [Trouble Shooting](#) (see page 125)
 - [User Interface is not responding any more \(client freeze\)](#) (see page 125)
 - [Find the problem on the client](#) (see page 125)
 - [Find the problem on the server](#) (see page 126)
 - [Connection timeouts between Server and Client](#) (see page 126)

- [Rich Text Editor](#) (see page 127)
 - [Unable to use the clipboard actions "Cut", "Copy" and "Paste"](#) (see page 127)
- [Workaround to prevent the "Invalid Certificate" error \(Mozilla\)](#) (see page 127)
- [Product Paradim \(EGD\) limitations](#) (see page 128)

3.1 Maintenance Tasks

- Backup strategy
- Workspace configuration

3.2 Tuning

3.2.1 Performance optimization of the "Documents" view

PIM - Server shows in its Documents view the content of a media asset category delivered by the integrated media asset provider.

However, the respective media asset providers vary in the data they deliver for a media asset and in the reaction time/speed.

For instance, it might be a problem to access a preview image at provider "A", or the calculation of an image dimension at provider "B" is a problem.

Therefore, the following requirement to media asset providers has been introduced:

- The media asset provider must deliver for each media asset an (unique) identifier, the name, and size in high speed.

If this requirement is not fulfilled, it makes no sense to integrate such a media asset provider.

Furthermore, the load of this view has been optimized so that it is based on the properties returned by the respective media asset provider:

When you are opening a media asset category, a list of all media assets contained by this category is requested from the media asset provider.

The media asset provider has to return the identifiers, the names, and the file sizes immediately.

The values for all other columns the media asset provider did not immediately return can be returned with a "delayed" status.

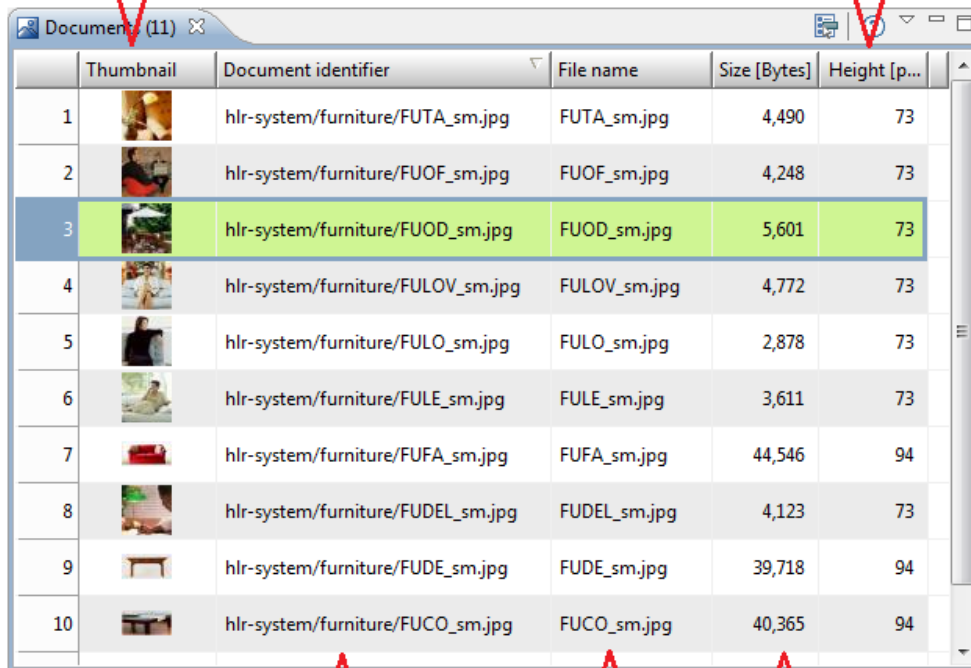
In this case, the view requests these values for each row separately, when the view "scrolls" in the visible field area.











The performance of the view can be improved by hiding unimportant columns.

It is also possible to hide the preview image column. This increases the performance even more.


asynchronous loading

asynchronous loading, if delayed by the media asset system



	Thumbnail	Document identifier	File name	Size [Bytes]	Height [p...
1		hlr-system/furniture/FUTA_sm.jpg	FUTA_sm.jpg	4,490	73
2		hlr-system/furniture/FUOF_sm.jpg	FUOF_sm.jpg	4,248	73
3		hlr-system/furniture/FUOD_sm.jpg	FUOD_sm.jpg	5,601	73
4		hlr-system/furniture/FULOV_sm.jpg	FULOV_sm.jpg	4,772	73
5		hlr-system/furniture/FULO_sm.jpg	FULO_sm.jpg	2,878	73
6		hlr-system/furniture/FULE_sm.jpg	FULE_sm.jpg	3,611	73
7		hlr-system/furniture/FUFA_sm.jpg	FUFA_sm.jpg	44,546	94
8		hlr-system/furniture/FUDEL_sm.jpg	FUDEL_sm.jpg	4,123	73
9		hlr-system/furniture/FUDE_sm.jpg	FUDE_sm.jpg	39,718	94
10		hlr-system/furniture/FUCO_sm.jpg	FUCO_sm.jpg	40,365	94

synchronous loading

 Note: Thumbnails of pdf documents are not displayed in the documents view!

3.2.1.1 Filter

The Documents view provides an interface enabling to filter the content on e.g. JPEG documents. To do this, you have to register an implementation of the Interface

```
com.heiler.ppm.mediaasset.ui.api.IFileDataFilter
```

by implementing the method

```
com.heiler.ppm.mediaasset.ui.views.DocumentsOfCategoryTableView.registerFileDataFilter(IFileDataFilter)
```

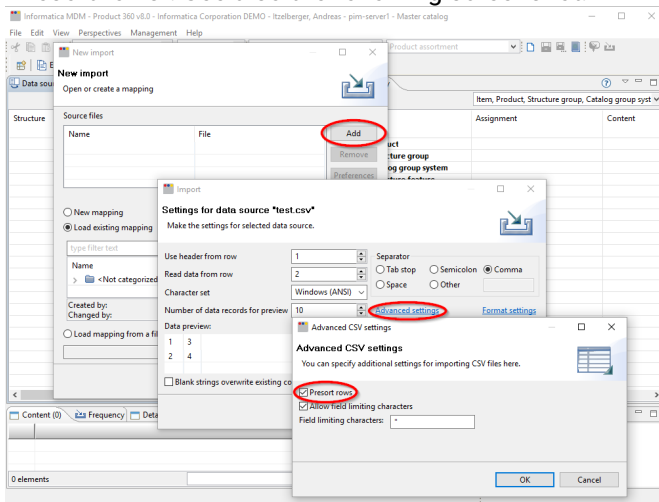
of the view. This filter will then be called every time the view is populated with data. From a technical point of view this filter will be called inside the respective IContentProvider implementation.

```
public interface IFileDataFilter
{
    public FileData[] filter( FileData[] fieldTofilter );
}
```

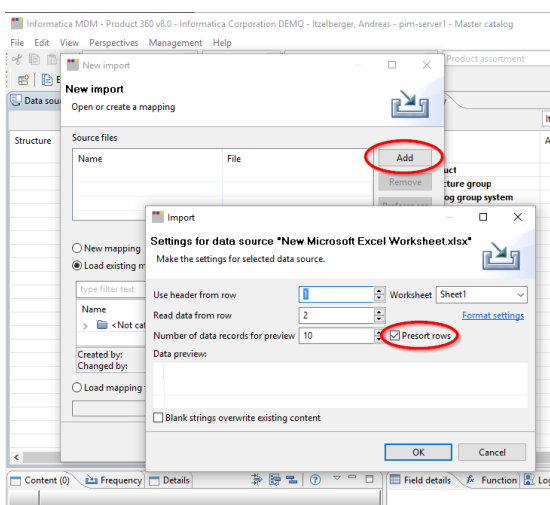
3.2.2 Import

When adding CSV or Excel files to an import project, per default all rows are read into heap memory in order to presort them. This speeds up processing if the data for one item is stored in rows that are spread. So when adding large CSV or Excel data files, consider to increase the client's heap size accordingly.

If presorting is not needed (because the data files are already presorted) or desired, it can be disabled by opening the "Advanced settings" dialog after selecting the CSV file, and here unchecking the checkbox "Presort rows". See also the following screenshot:



For Excel data files find the "Presort rows" checkbox directly in the Settings dialog, after selecting the Excel file:



This "Presort rows" settings only affects the adding of data files to the import project in P360 Desktop Client, but not the import itself. For disabling presorting rows in the import itself, please see the chapter "[Server Operation \(see page 42\)](#)" (of the Installation and Operation guide) and here the sub chapter "Import", "Preferences regarding memory usage".



Adding large data files to an import project might become slower, when presorting is deactivated (see setting above) and the import data file is not already presorted.

3.3 Trouble Shooting

- Log Files
- Workspace log
- Server connection lost/reconnect

3.3.1 User Interface is not responding any more (client freeze)

The currently supported operating systems of the Heiler Product Manager do only support a single GUI thread. It is responsible for reading and dispatching operating system events like mouse clicks and keyboard strokes. Usually the main thread reads an event from the operating system, and dispatches this event further to whoever listens on it (in our case SWT widgets). It then waits for all those event listeners to complete and then read again the next events from the operating system (the read-and-dispatch loop is synchronously). It performs this in an endless loop as long as the application is running.

In the PIM - Desktop this thread is also the `main` thread, which makes it quite easy to identify. Therefore, whenever the client stops responding it must be related to the client's `main` thread being busy with other things then reading and dispatching OS events. For example it's waiting on some operation to complete.

3.3.1.1 Find the problem on the client

To pinpoint the reason for the freeze you will need to know what the client is doing (or better waiting for) at the moment of the freeze. For this you will need a How to create a thread dump of the client.

1. Perform a thread dump of the client at the moment of the freeze
2. Find the `main` thread in the dump and check the stack trace - see where it ends.
3. In case the `main` thread waits on the completion of a server request you will need to move your investigation to the server.
In this case the stack trace of the client's `main` thread usually stops somewhere in the communication framework, e.g. `ClientNodeImpl(AbstractNode).sendRequest(NodeIdentifier, Message, long)`
4. In case the stack trace makes absolutely no sense for you, don't panic, that's not unusual 😊. With the information you collected we should be able to help you identify and fix the problem.
5. In case the client's main thread is waiting for a server request to complete, you can move on step further and check the server's threads.

3.3.1.2 Find the problem on the server

Since the server has no user interface, the main thread of the server is more or less irrelevant for this task. All requests from the communication framework are handled by the communication worker threads. The only problem is to find the right one. Currently you can't determine this by just looking at the thread names, you really have to check all communication framework threads. Communication worker threads are named `CommunicationWorker-<Number>`. Most of them should be in parking position, waiting for a request of a client (e.G. `Unsafe.park(boolean, long)` line: not available [native method]), those can be ignored, of course.



If you take multiple thread dumps of the server, you can check which thread "didn't move" between those dumps, making it easier to find the responsible one.

When you finally found the communication worker thread which is responsible, it might happen that this thread is itself waiting. Waiting for some other thread or the database. From there on it gets complicated and leads to debugging multi-threading issues and database locking problems. It would go beyond the scope of this wiki, so we would encourage you to contact Heiler Software Support if you need any further assistance on this.

3.3.1.3 Connection timeouts between Server and Client

In case there are connection issues between the server and the client regarding the heartbeat timeout it is possible to change the settings in the `plugin_customization.ini` of the client in the section "Communication CORE Settings".

If these settings do not generate the expected outcome it is also possible to change the rate in which the server checks for messages to the client. As the behavior is that if the server has nothing to send to the client in a specific time frame it will send a heartbeat object to the client. This time frame can be edited in the server's `plugin_customization.ini` if you add the following key:

```
com.heiler.ppm.communication.core/serverPollRate = 10
```

The value is the time in seconds which the server waits for a new message to send it to the client. Default value is 10.

When editing this value please keep in mind that it cannot be lower than the value of `heartbeatInterval`. Otherwise the heartbeat will fail as soon as no activity is send to the client. If the client do not get bytes in a specific time frame from the server it marks the connection as unusable and the client disconnects from the server.

To disable the heartbeat from client to server in general you can set the `heartbeatInterval` to 0 in the client `plugin_customization` and set the `serverPollRate` in the server `plugin_customization` to 0.

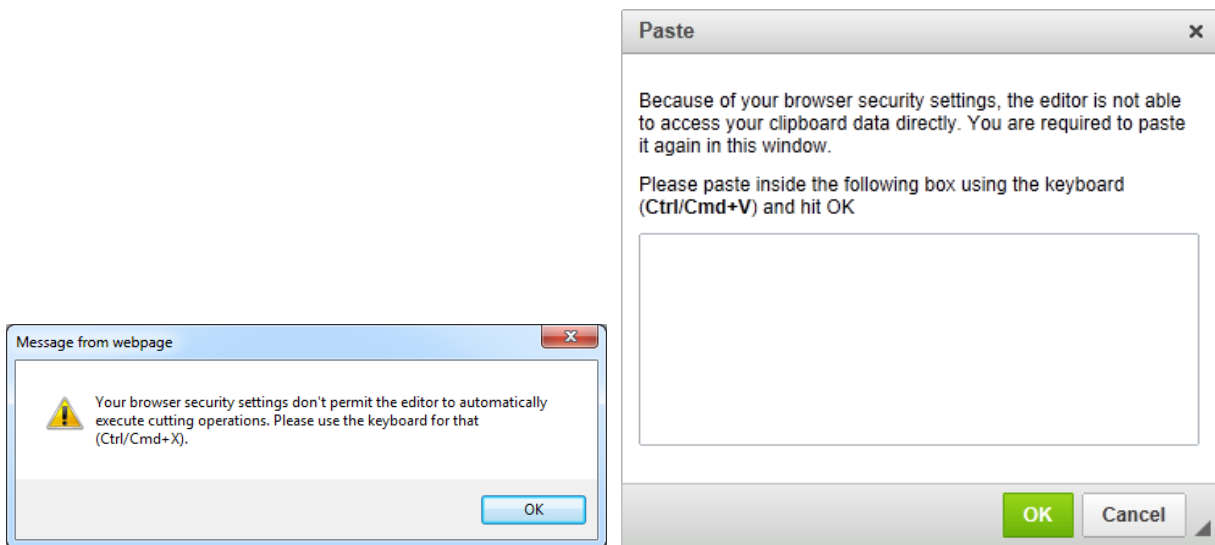
3.3.2 Rich Text Editor

3.3.2.1 Unable to use the clipboard actions "Cut", "Copy" and "Paste"

Due to the browser setting it may be impossible to use the following RichText-Editor buttons



In this case one of the following dialogs may be shown:

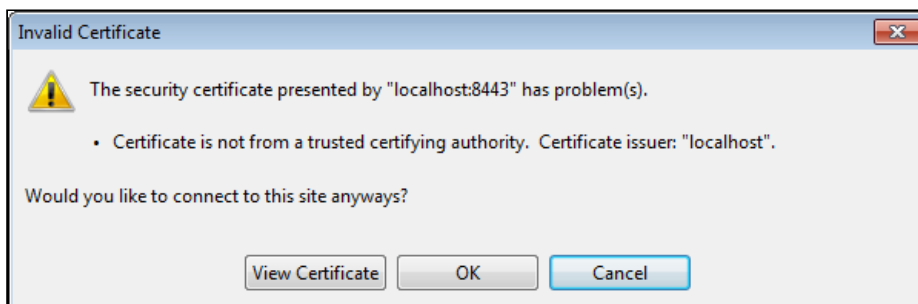


To change this behavior open the "Internet Explorer" and do following steps:

1. Go to Tools -> Internet Options.
2. Go to "Security" tab.
3. Select "Internet" zone, then click on "Custom level..." button.
4. Scroll down to "Scripting" section (at the bottom few).
5. Under "Allow Programmatic clipboard access" option, check or select (tick) to enable

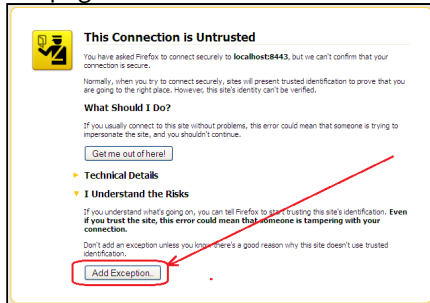
3.3.3 Workaround to prevent the "Invalid Certificate" error (Mozilla)

If the browser control loads any URL which uses an **unsigned** (or self-signed) SSL certificate, an error dialog appears:



Since there is no way to add an "exception" to the XULRunner, the following workaround can be used:

1. Open Firefox and load the domain which uses the invalid certificate (i.e. "https://localhost:8443/")
2. The page "This connection is untrusted" will be shown (see below):



3. Click on the button "Add exception..." and confirm the dialog which appears
4. Go to the "Firefox" profile folder (typically "C:\Users\[USER_NAME]\AppData\Roaming\Mozilla\Firefox\Profiles\[PROFILE_NAME]\")
5. Copy the file "**cert_override.txt**"
6. Go to the XULRunner profile folder (typically "C:\Users\[USER_NAME]\AppData\Roaming\Mozilla\eclipse\")
7. Paste the file "**cert_override.txt**"

Alternatively it is possible to use another workaround:

1. Open the file "prefs.js" in the folder "C:\Users\[USER_NAME]\AppData\Roaming\Mozilla\eclipse\".
2. Add the following line to the file and save the file:
`user_pref("network.automatic-ntlm-auth.trusted-uris", [host url of IdP server]);`

Next time you start PIM Desktop the browser control will load the URL without any errors.

Alternatively just use a "trusted" certificate (see https://en.wikipedia.org/wiki/Certificate_authority) or switch the setting `http.client.proxy` in "server.properties" to an HTTP URL (instead of HTTPS).

Starting with 8.0.03 it is possible to login to the desktop client via SAML. In this case the same rules apply to the certificate of the SAML IDP server. Generally it is recommended to use a trusted certificate, however if necessary the workaround with the `cert_override.txt` can be applied here as well.

3.4 Product Paradim (EGD) limitations

Please note that the EGD (Extended Generic Data) based product paradigm, which was used in the past for the first PIM Product paradigm, is now with PIM 7 no longer in Support. The EGD itself will remain as basis for custom entities, but without the `EGDAttributeMapping` entity.

This information is also about some sideeffects in the PIM 7 user interface, yet. For example you are having in the organization perspective double menu entries for product assortments. One is for the old, egd based, product assortment, one for the new product assortments. Unfortunately you cannot see the difference in one glance. Workaround until this is removed with PIM 8 is to try out default object rights on your product assortment. If it does not have the adjusted effect you have to choose the other selection in the default object rights combo box.

4 Audit Trail Operation

This page describes how to backup and restore audit trail data from and to elastic search indexes. It's typically used in environments in which the builtin backup/restore features of elastic can not be used or the audit trail documents should be transferred to a different kind of storage. The exported data is pure JSON and has no dependencies to elasticsearch any more. Scripts and examples of this page work for CentOS Linux environments.

For everyday backup tasks in an on-premises scenario we would recommend to use the out of the box features of elasticsearch. Please refer to the corresponding documentation of Elasticsearch for this.

In a standard operation mode of Product 360, the audit trail processor intercepts every data management request (create/update/delete) and generates an entity item change document JSON which is stored in the Elasticsearch server. If the Elasticsearch server is not accessible, then the transaction is rolled back in Product 360. There might be intermittent network failures between Elasticsearch and Product 360 servers, so there are in-built watchdog threads running in Product 360 on periodic intervals to maintain the eventual data consistency between the product data in Product 360 relational databases and audit trail data in Elasticsearch.

The audit trail storage lifecycle is powered by Elasticsearch's index lifecycle management. Depending on the retention policy defined for an entity in the repository, the lifecycle management is activated.

By default LONG_RETENTION will delete the audit trail data after 2 years and SHORT_RETENTION will delete the audit trail data after 2 months. The retention policies can be configured. Please refer to the configuration manual.



Customers should take great care in deciding how long they should retain their audit trail data.

4.1 Backup to File

It is always good to archive audit trail data in a human-readable format before it is purged. The following shell script helps in achieving the archival of audit trail data.

Install a third-party library called "Elasticdump"

```

1      * npm (specific to CentOS)
2      1. If npm already installed, goto elasticdump installation
3      2. If npm installed already and want to upgrade to latest version,
      goto step 4
4      3. If npm not installed, goto step 5
5      4. uninstall old version of npm
6          - sudo rm -rf /var/cache/yum
7          - sudo yum remove -y nodejs
8          - sudo rm /etc/yum.repos.d/nodesource*
9          - sudo yum clean all
10     5. install npm version (Example version 15)
```

```

11      - curl --silent --location https://rpm.nodesource.com/setup_15.x |
sudo bash -
12      6. install nodejs
13      - sudo yum -y install nodejs
14
15      * elasticdump
16      - npm install elasticdump@6.31.5 -g

```

audit_trail_backup.sh

```

1  #!/bin/bash
2  # Audit trail backup/archiving script
3  #####
4  # Script Name   : audit_trail_backup.sh
5  #
6  # Description   : Script for archival of elasticsearch audit trail data
7  #                 using elasticdump      #
8  # Args          : entityName, hasOwnIndex, datasource, startDate, endDate
9  #
10 #####
11 #####
12 #
13 # Configurable base parameters. Please refer README for usage
14 #
15 #####
16
17 declare -r ELASTIC_BASE_URL="http://localhost:9200"
18
19 declare -r BASE_DOWNLOAD_PATH="home/User/AUDITTRAILBACKUP"
20
21 declare -r P360_SYSTEM_NAME="localhost"
22
23 declare -r ELASTICDUMP_BATCH_SIZE=5000
24
25 declare -r ARCHIVE_FILE_SIZE_MBS=100
26
27 declare -r QUIET_MODE=false
28
29 #####
30 #####
31 # Mandatory parameters required to uniquely identify indexes and time
32 # range for archive #
33 # Please do not change them
34 #

```

```

30 #####
31 #####
32 declare entityName=${entityName}
33
34 declare datasource=${datasource}
35
36 declare hasOwnIndex=${hasOwnIndex}
37
38 declare startDate=${startDate}
39
40 declare endDate=${endDate}
41
42
43 while [ $# -gt 0 ]; do
44     if [[ $1 == *"--"* ]]; then
45         param="{1/--/}"
46         declare $param="$2"
47     fi
48     shift
49 done
50
51 #####
52 #####
53 ##
54
55 validate() {
56     if [ -z "$entityName" ] || [ -z "$endDate" ] || [ -z "$startDate" ] ||
57     [ -z "$datasource" ] || [ -z "$hasOwnIndex" ]; then
58         echo "validation failed, Please check mandatory fields and try
59         again!"
60         exit 0
61     fi
62     if [ -z "$(which elasticdump)" ]; then
63         echo "elasticdump not installed, Please refer README file for
64         installation details"
65         exit 0
66     fi
67 }
68
69 getElasticSearchIndexURL() {
70     # elastic indices are in lowercase
71     entityNameLowerCase=${entityName,,}
72
73     if [ $hasOwnIndex == "true" ]; then
74         local elasticSearchIndexName="audit_${entityNameLowerCase}_${
75         {datasource}*"

```

```

75         elasticSearchIndexURL="${ELASTIC_BASE_URL}/${P360_SYSTEM_NAME}.${
{elasticSearchIndexName}"
76
77         # entities that do not have their own indices are stored in elastic
shared indices
78         else
79             local elasticSearchIndexName="audit_*term_${datasource}";
80             elasticSearchIndexURL="${ELASTIC_BASE_URL}/${P360_SYSTEM_NAME}.${
{elasticSearchIndexName}"
81         fi
82
83         #validate elastic index
84         local httpStatusCode=$(curl -s -w "%{http_code}" -I
"$elasticSearchIndexURL?allow_no_indices=false" -o /dev/null)
85
86         if [ $httpStatusCode != "200" ]; then
87             echo "$elasticSearchIndexURL does not exists"
88             exit 0
89         fi
90
91         # exclude migrate indices from backup
92         elasticSearchIndexURL="$elasticSearchIndexURL,-${P360_SYSTEM_NAME}.${
{elasticSearchIndexName}_migrate"
93     }
94
95     getFolderName() {
96
97         local monthYear=${startDate:0:7}
98         folderName="audit_${entityNameLowerCase}_${datasource:-default}_${
{monthYear}"
99     }
100
101     createFolderIfNotPresent() {
102         mkdir -p "${BASE_DOWNLOAD_PATH}/${folderName}"
103     }
104
105     buildElasticSearchQuery() {
106
107         searchQuery="{\"query\":
108             {\"bool\":
109                 {\"filter\" :
110                     [{\"term\" : { \"_entity\" : \"$entityName\" } },
111                     {\"term\" : {\"_relationshipType\":
112                         \"changeSummaryDoc\" } },
113                     {\"range\": {\"_eventTimestamp\": {\"gte\":\"$
{startDate}\",\"lt\":\"${endDate}\"}}}}]}\"
114     }
115
116     executeElasticdumpCMD(){
        elasticdump --input=$elasticSearchIndexURL --searchBody="$searchQuery" --
output=$BASE_DOWNLOAD_PATH/$folderName/data.json --fileSize=$
{ARCHIVE_FILE_SIZE_MBS}mb --sourceOnly --quiet=$QUIET_MODE --
limit=$ELASTICDUMP_BATCH_SIZE --fsCompress

```

```

117 }
118
119 purgeEmptyDirs(){
120     if [ ! "$(ls -A $BASE_DOWNLOAD_PATH/$folderName)" ]; then
121         echo "$BASE_DOWNLOAD_PATH/$folderName is empty, going to purge"
122         rm -r $BASE_DOWNLOAD_PATH/$folderName
123     fi
124 }
125
126 ##### START #####
127
128 # validate user input
129 validate
130
131 # construct index complete path
132 getElasticSearchIndexURL
133
134 # download file folder name
135 getFolderName
136
137 # create download folder
138 createFolderIfNotPresent
139
140 buildElasticSearchQuery
141
142 # execute dump command
143 executeElasticdumpCMD
144
145 #delete empty folders
146 purgeEmptyDirs
147
148 ##### END #####

```

How to use audit_trail_backup.sh

Configure below attributes

- ELASTIC_BASE_URL : Elasticsearch url base path e.g. "http://localhost:9200"
- BASE_DOWNLOAD_PATH : directory to keep downloaded data e.g. "/home/Users/audittrailbackup"
- P360_SYSTEM_NAME : system name as defined in server properties, Example: "localhost"
- ELASTICDUMP_BATCH_SIZE: size of batch processed in one go, recommended value : 5000
- ARCHIVE_FILE_SIZE_MBS: max size of split files created in mbs e.g. 100
- QUIET_MODE : Set it to true to suppress elasticdump logs

This **file** is called from wrapper.sh generally, but it could also be called independently using below **command format**

```
bash audit_trail_backup.sh --entityName "Article" --hasOwnIndex "true" --
datasource "master" --startDate 01-May-2020T00:00:00 --endDate 31-May-2020T23:59:59
```

In order to archive multiple entities, the following shell scripts can be used.

wrapper.sh

```

1  #!/bin/bash
2
3  #####
4  # Script Name   : wrapper.sh
5  #
6  # Description   : wrapper script to call backup script by providing
7  #                 required arguments
8  # Args          : User has to provide entityName, datasource and
9  #                 hasOwnIndex arguments by checking repository settings
10 # Author        :
11 #
12 # Version       : baseline
13 #
14 #####
15 #####
16 #
17 #                 **IMPORTANT**
18 #
19 # User has to provide list of "{entityName}, {datasource}, {ownIndex}" as
20 # it is from repository #
21 # for all entities which require backup
22 #
23 #####
24 #####
25 ENTRIES=(
26     "Article, master, true"
27     "Article, supplier, true"
28     "Variant, master, true"
29     "Variant, supplier, true"
30     "Product2G, master, true"
31     "Product2G, supplier, true"
32     "StructureGroup, main, false"
33     "Structure, main, false"
34 )

```

```

29
30
31 TRACKER_FILE_BASE_PATH=~ /AUDITTRAILBACKUP
32
33 BACKUP_SCHEDULE_MONTHS=12
34
35 #####
36 #####
37 # Used to get time range for the data to be backed up. This is calculated
38 # by subtracting #
39 # BACKUP_SCHEDULE_MONTHS from current month. This is to be specified by
40 # the user. #
41 # Example - For BACKUP_SCHEDULE_MONTHS = 6 and current month Nov'20
42 #
43 # backup start date is 01-May-2020 00:00:00 and end date 31-May-2020
44 # 23:59:59 #
45 #####
46 #####
47
48 backupMonthNumber=$(date -d "$(date +%m) - $BACKUP_SCHEDULE_MONTHS month"
49 '+%0m')
50
51 backupYear=$(date -d "$(date +%Y-%m-1) - $BACKUP_SCHEDULE_MONTHS month" +%Y)
52
53 backupMonthDays=$(cal $backupMonthNumber $backupYear | xargs echo | awk
54 '{print $NF}')
55
56 backupStartDate=$(date -d "$(date +%Y-%m-1) - $BACKUP_SCHEDULE_MONTHS
57 month" +%Y-%m-%d'T'00:00:00)
58
59 backupEndDate=$(date -d "$(date +%Y-%m-1) - $BACKUP_SCHEDULE_MONTHS month +
60 $(( $backupMonthDays - 1 )) days" +%Y-%m-%d'T'23:59:59)
61
62 ##### Creates separate logging folder#####
63 declare executionDateTime="$(date '+%Y-%m-%dT%H:%M:%S')"
64 mkdir -p logs/$executionDateTime
65
66 #####
67 #####
68 # Iterating over all entries and calling backup script for each
69 # entity,datasource,ownIndex #
70 # combination. #
71 #####
72 #####
73
74 for index in "${ENTRIES[@]}; do
75
76     IFS=' ' read -r -a entry <<< "$index"
77     entityName=${entry[0]}

```

```

67     datasource=${entry[1]}
68     hasOwnIndex=${entry[2]}
69     (
70         echo "backup script executing for entity: $entityName, datasource:
        $datasource, hasOwnIndex: $hasOwnIndex, startDate: $backupStartDate,
        endDate: $backupEndDate"
71
72         bash audit_trail_backup.sh --entityName $entityName --hasOwnIndex
        $hasOwnIndex --datasource $datasource --startDate $backupStartDate --
        endDate $backupEndDate
73
74         echo -e $entityName '\t' '\t' $datasource '\t' '\t' $hasOwnIndex '\t'
        '\t' $(date) >> $TRACKER_FILE_BASE_PATH/tracker.log
75     ) 2>&1 | tee -a ./logs/$executionDateTime/audit_${entityName,,}_$
        {datasource}.out
76
77     done

```

How to use wrapper.sh

wrapper.sh - wrapper **for** audit_trail_backup script

Configure below attributes

- ENTRIES : User has to provide list of "{entityName}, {datasource}, {ownIndex}" as it is from repository

for all entities **which** require backup

Example:

```

("Article, master, true"
 "Article, supplier, true"
 "Structure, main, false")

```

- TRACKER_FILE_BASE_PATH : Example: /home/Users/tracker

- BACKUP_SCHEDULE_MONTHS : Specify backup duration e.g. - **for** backup schedule of 6 and current month November'2020 it will archive whole of May'2020

bash wrapper.sh

Periodic backup

The wrapper.sh can be associated with a monthly cron job and every month audit trail data will keep getting archived.

4.2 Restore from File

Once the audit trail data is archived, it can be safely stored elsewhere. It is also easy to restore the audit trail data from the archived format. The following shell scripts help in restoring the archived audit trail data into a different Elasticsearch index.

restore_audit_trail_backup.sh

```

1  #!/bin/bash
2
3  #####
4  # Script Name   : restore_audit_trail_backup.sh
5  #
6  # Description   : restores archived files into elasticsearch
7  #
8  # Args          : archiveFolder path and index name
9  #
10 #####
11
12 ###Configurable base parameters.Please refer README for usage###
13 declare -r ELASTIC_BASE_URL="http://localhost:9200"
14
15 declare -r ELASTICDUMP_BATCH_SIZE=5000
16
17 declare -r INDEX_MAPPING_FILE="./elastic_index_mapping.json"
18
19 declare -r QUIET_MODE=false
20
21 # Mandatory parameters required to loads backup to elastic indices
22 declare archiveFolder=${archiveFolder}
23
24 declare index=${index}
25
26 while [ $# -gt 0 ]; do
27     if [[ $1 == *"--" ]]; then
28         param="{1/--/}"
29         declare $param="$2"
30     fi
31     shift
32 done
33
34 #####
35 # Index creation using json mapping present in INDEX_MAPPING_FILE
36 #
37 #####

```

```

34 declare value=$(tr -d '\040\011\012\015' < $INDEX_MAPPING_FILE)
35 curl -XPUT $ELASTIC_BASE_URL/$index -H 'Content-Type: application/json' -d
   $value
36
37 #####
38 # Iterate over .json extension files under archiveFolder, and
39 # loads data from each file to elastic index
40 #####
41 echo "Starting restoration from $archiveFolder to $index"
42 for filename in $archiveFolder/*.json.gz; do
43     (
44         echo $filename
45         gunzip -c $filename > $archiveFolder/restoration_intermediary.json
46         elasticsearchdump --input=$archiveFolder/restoration_intermediary.json
         --output=$ELASTIC_BASE_URL --output-index=$index --type=data --transform="
         doc._source=Object.assign({},doc)" --limit=$ELASTICDUMP_BATCH_SIZE --
         quiet=$QUIET_MODE
47         rm -rf $archiveFolder/restoration_intermediary.json
48     ) 2>&1 | tee -a log_restore_"$(date +%Y-%m-%d)".out
49 done
50 echo "Restoration complete"
51
52 ##### END #####

```

Audit Trail Mapping JSON

```

1  {
2    "mappings": {
3      "dynamic": "strict",
4      "properties": {
5        "_changeSummary": {
6          "type": "object",
7          "enabled": false
8        },
9        "_changeType": {
10         "type": "keyword"
11       },
12       "_changedEntities": {
13         "type": "keyword"
14       },
15       "_changedFields": {
16         "type": "keyword"
17       },
18       "_container": {
19         "properties": {
20           "_entityId": {

```

```

21         "type": "integer"
22     },
23     "_externalId": {
24         "type": "keyword"
25     },
26     "_internalId": {
27         "type": "keyword"
28     }
29 }
30 },
31 "_entity": {
32     "type": "keyword"
33 },
34 "_entityItem": {
35     "properties": {
36         "_entityId": {
37             "type": "integer"
38         },
39         "_externalId": {
40             "type": "keyword"
41         },
42         "_internalId": {
43             "type": "keyword"
44         }
45     }
46 },
47 "_eventTimestamp": {
48     "type": "date",
49     "format": "strict_date_optional_time_nanos"
50 },
51 "_identifier": {
52     "type": "keyword"
53 },
54 "_invalidReason": {
55     "type": "keyword"
56 },
57 "_migrationId": {
58     "type": "keyword"
59 },
60 "_module": {
61     "type": "keyword"
62 },
63 "_relationshipType": {
64     "type": "join",
65     "eager_global_ordinals": true,
66     "relations": {
67         "changeSummaryDoc": "triggerFiredDoc"
68     }
69 },
70 "_revision": {
71     "properties": {
72         "_entityId": {
73             "type": "integer"

```

```

74         },
75         "_externalId": {
76             "type": "keyword"
77         },
78         "_internalId": {
79             "type": "keyword"
80         }
81     },
82     "_transactionStatus": {
83         "type": "keyword"
84     },
85     "_triggerStatus": {
86         "type": "keyword"
87     },
88     "_user": {
89         "properties": {
90             "_entityId": {
91                 "type": "integer"
92             },
93             "_externalId": {
94                 "type": "keyword"
95             },
96             "_internalId": {
97                 "type": "keyword"
98             }
99         }
100     }
101 }
102 }
103 }
104 }

```

How to use restore_audit_trail_backup.sh

restore_audit_trail_backup.sh - loads data from backup json files to elastic indices

Configure below attributes

- ELASTIC_BASE_URL : Example: "http://localhost:9200"
- ELASTICDUMP_BATCH_SIZE: size of batch processed **in** one go, recommended value : 5000
- QUIET_MODE : Set it to **true** to suppress elasticdump logs
- DEFAULT_INDEX_MAPPING_FILE : Provide path to default P360 index mapping json **file** e.g " /home/Users/mapping.json"

```
bash restore_audit_trail_backup.sh --archiveFolder /home/
audit_product2g_master_2020-10 --index restored_audit_product2g
```

5 Log File Overview

This page gives an overview about the log files and their configuration options.

Module	Default Log File Location	Log File Configuration	Remarks
Application Server Web Client	<pre><ServerInstallationRoot>\server\logs*.log <ServerInstallationRoot>\server\wrapper.log</pre>	<pre><ServerInstallationRoot>\server\configuration\HPM\log4j2.xml</pre>	Application log files (out, error, perf) Windows Service Wrapper log
Desktop Client	<pre><ClientInstallationRoot>\client\logs*.log</pre>	<pre><ClientInstallationRoot>\client\configuration\HPM\log4j2.xml</pre>	
Supplier Portal	<pre><SupplierPortalInstallationRoot>\logs <SupplierPortalInstallationRoot>\tomcat\logs</pre>	<pre><SupplierPortalInstallationRoot>\configuration\logback.xml</pre>	Application Logs Tomcat Logs

Module	Default Log File Location	Log File Configuration	Remarks
Media Manager	<pre><MediaManagerInstallationRoot>\Informatica Media Manager\IMM_protocols <MediaManagerInstallationRoot>\OpasGWebServer\Tomcat\logs <MediaManagerInstallationRoot>\OpasGWebServer\XOBSessionManager\logs</pre>		
Web Search	<pre><WebSearchInstallationRoot>\log\server.log <WebSearchInstallationRoot>\<tomcat-server>\logs</pre>	<pre><WebSearchInstallationRoot>\<tomcat-server>\lib\log4j.xml <WebSearchInstallationRoot>\<tomcat-server>\conf\logging.properties</pre>	Application Logs Tomcat Logs
Database Setup	<pre><DatabaseSetupInstallationRoot>\dbclient\logs*.log</pre>	<pre><DatabaseSetupInstallationRoot>\dbclient\configuration\log4j2.xml</pre>	

5.1 Additional Logging of Dataquality

If you want verbose logging of the execution of a dataquality rule by the rule engine, go to the `log4j2.xml` file and change the priority of the IDQ category like below:

```
<Logger name="IDQ" level="DEBUG" />
```

The log files can also be found in the server log folder <ServerInstallationRoot>\server\logs\. Each rule configuration will be executed in it's own worker thread, so a log file will have following name pattern:

```
<JobWorkerName>-<RuleConfigurationName>-<StartDate>.log
```

5.2 Additional Logging of Change Summary and Datagraph

To get the insides of datagraphs and change summaries before store them to the database you can enable the following logger. You can define the logger for each subclass of

```
com.heiler.ppm.std.server.entity.EntityPersistenceManagerImpl
```

This sample enables the logging for `ArticlePersistenceManager`.

```
<Logger
name="com.heiler.ppm.article.server.internal.entity.ArticlePersistenceManager"
level="TRACE" />
```

Use this logger wisely. The logging output is big.

5.3 Additional Logging of Media Asset Processing

If you want to fetch the logging of JMS processing, go to the `log4j2.xml` file and add the following setting:

```
<Logger name="PPM.MEDIA_ASSET_NOTIFICATION" level="DEBUG"/>
```

If you want to fetch the logging of Media Manager connector API methods, go to the `log4j2.xml` file and add the following setting:

 Note: Connector logging via PIM is available since PIM 8.0.01

```
<Logger name="com.heiler.imm.connector" level="DEBUG" />
```

For more details information please visit the page [Logging for Media Manager Connector within PIM](#).

Copyright

© Copyright Informatica LLC 1993, 2024

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica LLC.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Informatica and the Informatica logo are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at <https://www.informatica.com/trademarks.html>. Other company and product names may be trade names or trademarks of their respective owners.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at infa_documentation@informatica.com.

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.