KPI Metrics Configuration Guide

An Open Source Asset for use with TIBCO® Data Virtualization

TIBCO Software empowers executives, developers, and business users with Fast Data solutions that make the right data available in real time for faster answers, better decisions, and smarter action. Over the past 15 years, thousands of businesses across the globe have relied on TIBCO technology to integrate their applications and ecosystems, analyze their data, and create real-time solutions. Learn how TIBCO turns data—big or small—into differentiation at www.tibco.com.

Project Name	AS Assets KPI Metrics
Document Location	This document is only valid on the day it was printed. The source of the document will be found in the ASAssets_KPI folder (https://github.com/TIBCOSoftware)
Purpose	Self-paced instructional

www.tibco.com

Global Headquarters
3303 Hillview Avenue
Palo Alto, CA 94304

Tel: +1 650-846-1000
+1 800-420-8450
Fax: +1 650-846-1005

Revision History

Version	Date	Author	Comments
1.0	Oct 2012	Manny Bhatia	Initial revision
1.4	Nov 2012	Matthew Lee	Minor updates to deployment language to clarify steps
1.5	Nov 2012	Matthew Lee	Revised setup and configuration steps for clarification
1.6	Sep 2013	Matthew Lee	Revised document for KPImetrics version 2.0
1.7	Jan 2014	Matthew Lee	Migrated to Cisco documentation standards
1.8	Aug 2017	Mike Tinius	Upgraded to work with Data Virtualization metrics with Postgres. Add-on to DV metrics capability.
1.9 – 2017.4	Dec 2017	Mike Tinius	Add additional capabilities and migrated to Tibco documentation standards
2018.1	March 2018	Mike Tinius	Optimization of Oracle and SQL Server views to achieve better push-down. Improved installation scripts.
2018.101	March 30 2018	Mike Tinius	Fixed an issue with the installation scripts. Added KPImetrics_worksheet.xlsx.
2018.102	April 23 2018	Mike Tinius	Added ldap and metrics history “userkey” field to allow normalization of the join key. Fixed an issue with CPUMemChecker.
2018.103	May 2 2018	Mike Tinius	Modified data transfer procedure. Added column CURRENT_OPERATION to METRICS_JOB_DETAILS table. Deprecated cache_status and cache_tracking usage.
2018.3	Sep 18 2018	Mike Tinius	Modified to allow datasource names in the format of KPI_oracle_11g, KPI_oracle_12c, KPI_sqlserver_2012 or KPI_sqlserver_2014 in order to allow for adding a datasource. Modified sqlParserV1_2 to allow for parallel processing of SQL when a cluster is present. A node can process rows other than its own rows when it has no work to do. Modified tables: METRICS_ALL_RESOURCES, METRICS_SQL_REQUEST Added tables: METRICS_SQL_CONTROL, METRICS_SQL_CONTROL_LOG Modified the native SQL data transfer script to use stage tables resuling in only inserts to the history tables to improve efficiency. Modified /shared/ASAssets/KPImetrics/Business/Logical/metrics *_hist views to use where starttime > TO_TIMESTAMP('1900-01-01 00:00:00.000') to force the use of parallel queries.
2018.301	Oct 2 2018	Mike Tinius	Modified parseSqlScriptComplex and parseSqlScriptTemplate to change template select from SELECT DISTINCT to SELECT TOP 1 to make it more efficient.
2018.302	Oct 12 2018	Mike Tinius	Fixed a collection table row volume problem by adding an independent delete trigger for the collection tables: metrics_requests and metrics_resources_usage.
2018.400	Dec 1 2018	Mike Tinius	Added processing for SQL request datasource lineage thus allowing reporting on what physical data source resources are related to which published resources. Requires Utilities 2018Q4.
2018.401	Dec 3 2018	Mike Tinius	Removed all reference and processing for METRICS_RESOURCES_USAGE_UD. This is essentially a copy of metrics_resources_usage_hist where resourcekind=’user defined’. This is a simplification.
2019.100	Jan 7 2019	Mike Tinius	Fixed installation script. Added “domainkey” to METRICS_LDAP_PERSON and history tables to qualify the “user”. The same user can be in multiple domains.
2019.101	Jan 30 2019	Mike Tinius	Added support for DV 8.0 as the out-of-the-box metrics tables [metrics_requests and metrics_sessions] added user, domain and

© Copyright 2000-2017 TIBCO Software Inc.

			group. KPI installation automatically checks for DV version 7.0 or 8.0 and executes the appropriate scripts to install KPImetrics.
2019.102	Mar 12 2019	Mike Tinius	Added upgrade scripts. Added “group” to metrics_requests and metrics_sessions for 7.0 to bring up to par with 8.0.
2019.200	Jun 17 2019	Mike Tinius	Migrated severl procedures to the Utilities. Requires Utilities 2019Q200. Put explicit BEGIN/END blocks around updates in pPurgeData. sqlParserV1_2 - Converted all VECTORS to XML as VECTOR parameters has been associated with memory leaks.
2019.300	August 5 2019	Mike Tinius	Fixed bug where pRebuildIndexes fails. Fixed with admin interface invocation. Added trigger kpimetricsTrig_24_ValidateUpdateStatusTables. Added procedure pValidateUpdateStatusTables.
2019.301	August 31 2019	Mike Tinius	Added KPI Metadata feature for capturing resource metadata and reporting. Fixed various bugs.
2019.400	October 24 2019	Mike Tinius	Fixed bugs with installation and SQL Server and Oracle DDL scripts.
2019.401	November 6, 2019	Mike Tinius	Various Metadata capability changes.
2019.402	December 23, 2019	Mike Tinius	Better debug and error handling. Data transfer script batching. Upgrade script modifications. Consolidated all customizable resources into /shared/ASAssets/KPImetrics/Customize.
2020.100	January 23, 2020	Mike Tinius	Created a data dictionary document “KPImetrics Data Dictionary v1.1.pdf”. Added transaction blocks in pPARTITION_MANAGER_exec, pqPartition_metrics_history_tables_ADD, and pqPartition_metrics_history_tables_DROP
2020.101	February 25, 2020	Mike Tinius	KPI Metadata improvements. Purge concurrency improvements.
2020.200	March 12, 2020	Mike Tinius	Includes 2020.102 + Requires Utilities 2020Q200 to synchronize with migrated utilities required by KPImetrics.
2020.201	April 6, 2020	Mike Tinius	Cluster bug fixes. Metadata bug fixes.
2020.202	May 1 2020	Mike Tinius	Removed Archive views by adding partitioning to hold current and history in the same table. Supporting SQL Server 2016 or higher.

Related Documents

Name	Version
How To Use Utilities.pdf	2020Q200
KPImetrics Overview.pdf	2020Q201
KPI Metrics Overview.pptx	2020Q202
KPImetadata Configuration Guide vX.Y.pdf	2020Q202
KPImetrics Data Dictionary vX.Y.pdf	2020Q202
KPImetrics_Table_Relationship_Diagram.pptx	2020Q202

Supported Versions

Name	Version
TIBCO® Data Virtualization	7.0.8 or later
AS Assets Utilities open source	2020Q200 or later

© Copyright 2000-2017 TIBCO Software Inc.

© Copyright 2000-2017 TIBCO Software Inc.

Table of Contents

1 Introduction ... 8
Purpose ... 8
Audience ... 8
References ... 8
Overview ... 8

2 Requirements ... 9

3 New Installation and Configuration ... 14
Supported Database Platforms ... 14
Installing KPImetrics ... 14
Turn off DV metrics ... 14
Download and Import the KPImetrics components to your DV instance ... 14
Configuration Overview ... 16
[1.] Configure the KPI_DV_AdminInterface data source ... 16
[2.] Configure the KPImetrics data source ... 17
[3.] Configuration Parameters ... 17
[4.] Configure Common Values ... 19
[5.] Configure Metrics Job Lookup Tables ... 24
[6.] Configure KPImetrics Triggers ... 26
[7.] Deploy CPU and Memory Checker shell scripts (Windows or UNIX) ... 30
[8.] Configure LDAP ... 30
[9.] Configure Metadata ... 33
Execute Post-Installation Script ... 33
Execute Installation Script ... 34
Deploy CPU and Memory Checker shell scripts (Windows and UNIX) ... 36
Enable Triggers ... 37
Configure DV Out-of-the-box Metrics ... 40

4 Upgrading KPImetrics ... 43
Introduction ... 43
How to Upgrade KPImetrics ... 43
Requirements: ... 43
Migrating from 7.x to 8.x: ... 43
Current Version: ... 43

5 Migrate a New KPImetrics Installation ... 55
Export Source Environment .car file ... 55
Import Target Environment .car file ... 57
Configure KPImetrics Target Installation ... 59
Execute KPImetrics Installation script ... 61
Configure TDV Metrics [/policy/metrics] ... 63
Turn on KPImetrics triggers ... 63

6 KPImetrics Administration Scenarios ... 64
Turn KPI On/Off ... 64
Turn Data Virtualization (DV) metrics On/Off ... 66
Modify Triggers ... 66
Perform Oracle Database Maintenance on Collection Tables ... 68

© Copyright 2000-2017 TIBCO Software Inc.

Perform SQL Server Database Maintenance on Collection Tables ... 68
Configure Third Party Tool Access ... 70
Get the Current Row Distribution for the History Tables/Partitions ... 70
FAQ 71

7 KPImetrics Resources ... 73

Configuration Resources ... 73
KPI Version Overview ... 73
Configuration Customize Folder Overview ... 73
Configuration Folder Overview ... 74
Published Resources ... 77
KPImetricsAdmin Definitions ... 77
configurations.updateTriggers Procedure ... 77
Data Sources ... 77
Metadata Data Source for LDAP ... 77
Metadata Data Source for CPUAndMemChecker ... 78
Metadata Data Source Tables and Procedures ... 79
KPImetrics Metrics Table Relationship Diagram ... 80
KPImetrics Supporting Table Relationship Diagram ... 80
KPImetrics Metadata Table Relationship Diagram ... 81
KPI Metrics Metadta Partitioning Strategy (w/leap year) ... 81
KPI Metrics Metadta Partitioning Strategy (w/no leap year) ... 82
Metadata System Triggers and Load Scripts ... 92
Metadata System Helpers Scripts ... 106
Baseline Table Load Architecture ... 108
Baseline Table Accessibilty and Parallel Table Load Architecture ... 108
Data Transfer Scripts ... 110
Physical Oracle Data Transfer Script ... 110
Physical SQL Server Data Transfer Script ... 118

8 Release Notes ... 125

Added or Modified in this Release ... 125
Release 2020Q202 [May 1 2020] ... 125
Release 2020Q201 [Apr 6 2020] ... 125
Release 2020Q200 [Mar 12 2020] ... 126
Release 2020Q101 [Feb 25 2020] ... 128
Release 2020Q100 [Jan 23 2020] ... 132
Release 2019Q402 [Dec 23 2019] ... 133
Release 2019Q401 [Nov 6 2019] ... 137
Release 2019Q400 [Oct 24 2019] ... 139
Release 2019Q301 [Aug 31 2019] ... 140
Release 2019Q300 [Aug 5 2019] ... 142
Release 2019Q200 [Jun 13 2019] ... 142
Release 2019Q102 [Mar 12 2019] ... 143
Release 2019Q101 [Jan 30 2019] ... 144
Release 2019Q100 [Jan 7 2019] ... 145
Release 2018Q401 [Dec 4 2018] ... 146
Release 2018Q400 [Dec 1 2018] ... 147
Release 2018Q302 [Oct 12 2018] ... 148
Release 2018Q301 [Oct 1 2018] ... 148
Release 2018Q3 [Sep 2018] ... 148

9 Appendix A – Partitioning Schemes ... 151

© Copyright 2000-2017 TIBCO Software Inc.

Oracle Partition Scheme 151
SQL Server Partition Scheme 157

10 Post-Installation Background Information 163

Information Only Section 163
Create the KPImetrics storage tables for Oracle 164
Create the KPImetrics storage tables for SQL Serer 166
Common Configuration for all Databases 168

© Copyright 2000-2017 TIBCO Software Inc.

1 Introduction

Purpose

The purpose of this document is to provide guidance on how install, monitor and use the AS Assets KPI Metrics.

Audience

This document is intended to provide guidance for the following users:

- Data Virtualization Administrators – provides a guide for installation.
- Architects – provides the KPImetrics architecture.
- Data professionals – provides background on the published views and usage.
- Operations users – provides insight into triggers and procedures that are executed.
- Project Managers – provides general information on KPImetrics.

References

Product references are shown below. Any references to CIS or DV refer to the current TIBCO® Data Virtualization.

- TIBCO® Data Virtualization was formerly known as
    - Cisco Data Virtualization (DV)
    - Composite Information Server (CIS)

Overview

Please review the document "**KPImetrics Overview.pdf**".

© Copyright 2000-2017 TIBCO Software Inc.

2 Requirements

The following requirements and pre-requisites must be met:

- **Minimum** of **Data Virtualization (DV) 7.0.8.00 HF01** is recommended as there have been DV Metrics bug fixes.
- TIBCO Open Source **ASAssets Utilities_2020Q200.zip** is required as a baseline. However, it is recommended to download whatever the "**latest release**" is. Follow the instructions within the zip file for installation.
- Use **KPImetrics_worksheet.xlsx** as a way to gather the required information for installation.
- Acquire the LDAP connection information and credentials that will be used to setup the KPImetrics LDAP data source.
- A DV admin user will be created at installation time as "metrics_app_id@composite" so that resource ownership can be assigned to all /shared/ASAssets/KPImetrics and /services/databases/ASAssets/KPImetrics. Required for metrics data filter deletion.
    - Upon create the user studio rights are set as follows: ACCESS_TOOLS, READ_ALL_CONFIG, READ_ALL_RESOURCES, READ_ALL_STATUS, READ_ALL_USERS, MODIFY_ALL_CONFIG, MODIFY_ALL_RESOURCES
        - MODIFY_ALL_CONFIG and MODIFY_ALL_RESOURCES are required for Cache_METADATA_TABLES to read row-based security and column-based security metadata.
- Database Requirements
    - Oracle 11g, 12c or later (recommended due to better push-down capabilities)
        - Requires partitioning.
    - SQL Server 2016 or higher or SQL Server Azure
        - Requires partitioning + syntax: truncate table <t> with(partitions(n))

© Copyright 2000-2017 TIBCO Software Inc.

- ***Previous implementations of KPImetrics using SQL Server 2012/2014 must migrate their database to SQL Server 2016.***

- **Data rention**: The amount of time to retain the data before purging. The period suggestions are as follows:
  - Lower-level environments (LLE) – 4 months for LLE.
  - Production (PROD) – 13 months in production PROD.
    - Metrics history partitioning uses a month strategy so 13 is recommended as partitions are dropped based on a month. The Metadata uses a 366-day interval partitioning strategy so it can be up to 366 days of retention and uses a sliding window where partitions are truncated but not dropped.
- **Storage**: Minimum of 100 GB to start with and be prepared to grow.
  - **Oracle Tablespaces**:
    - *Logging*: The objective is to keep logging to a minimum especially in LLE. As described below the collection tables [METRICS_DATA_COLL] should always be configured for NOLOGGING in all environments.
    - *Tablespace*=METRICS_DATA_COLL
      - This will contain the DV metrics collection tables [metrics_sessions, metrics_requests, metrics_resources_usage]
      - Configured to be a **NOLOGGING** tablespace so that it is more efficient for providing insert/delete operations every 2 hours without impacting performance. Rows will be transferred via to the history tables which are stored in the tablespace METRICS_DATA.
      - Initial 20 GB
    - *Tablespace*=METRICS_DATA_HIST
      - This will contain the KPImetrics tables.
      - Configured with logging and large enough to hold 100 GB and grow as needed.
      - Initial 40 GB
    - *Tablespace*=METRICS_DATA_IDX
      - This will contain the KPImetrics indexes.

© Copyright 2000-2017 TIBCO Software Inc.

        - Configured with logging and large enough to hold 100 GB and grow as needed.
        - Initial 40 GB
    - **SQL Server File Groups**:
        - *Logging*: The objective is to keep logging to a minimum especially in LLE. For LLE environments the database should be set to "Recovery mode"=SIMPLE.
        - *Filegroup*=METRICS_DATA_COLL
            - This will contain the DV metrics collection tables [metrics_sessions, metrics_requests, metrics_resources_usage]
            - Data will be processed every 2 hours. Rows will be transferred via to the history tables which are stored in the tablespace METRICS_DATA.
            - Initial 20 GB
        - *Filegroup*=METRICS_DATA_HIST
            - This will contain the KPImetrics tables.
            - Configured with logging and large enough to hold 100 GB and grow as needed.
            - Initial 40 GB
        - *Filegroup*=METRICS_DATA_IDX
            - This will contain the KPImetrics indexes.
            - Configured with logging and large enough to hold 100 GB and grow as needed.
            - Initial 40 GB
- **Database and schema**: A separate database for each environment is strong recommended. This is due to the "intensity" of collection and the amount of data that gets collected. Additionally, it makes it easier to migrate the TDV code from one environment to the next when upgrades are required.
    - A database schema is required for each TDV environment to store the Metrics data for a period of time specified by the user. **KPImetrics cannot support 2 or more different TDV environment instances in the same schema**. For a TDV cluster, the same database schema "must" be used.
    - It is highly recommended to use the same catalog (dbname) and shema for each environment to make TDV code migration easier.

© Copyright 2000-2017 TIBCO Software Inc.

- Recommended schema (user) or **database name=DVKPI**

- **Oracle Privileges**: Must be granted privileges for the schema user:
  - SELECT/INSERT/UPDATE/DELETE TABLE
  - CREATE/DROP/ALTER TABLE
  - CREATE/DROP/ALTER INDEX
  - EXECUTE/CREATE/DROP PROCEDURE
  - SELECT/CREATE/DROP SEQUENCE
  - Oracle specific:
    - CREATE/ALTER SESSION
    - RESOURCE
    - EXECUTE SYS.DBMS_STATS.GATHER_TABLE_STATS
    - CREATE/DROP/ALTER PARTITION
      - ALTER TABLE ADD PARTITION
      - ALTER TABLE DROP PARTITION
    - SELECT SYS.ALL_TAB_PARTITIONS

- **SQL Server Privileges**: Must be granted privileges for the schema user:
  - Option 1 – General access to the DV KPImetrics DB user:
    - Give the SQL Server user "dbadmin" +

      ```
      GRANT EXECUTE ON SCHEMA::[<schema>] TO <username>
      ```

  - Option 2 – Granular privileges:
    - SELECT/INSERT/UPDATE/DELETE TABLE
    - CREATE/DROP/ALTER TABLE
    - CREATE/DROP/ALTER INDEX
    - EXECUTE/CREATE/DROP PROCEDURE
    - SELECT/CREATE/DROP SEQUENCE
    - SQL Server specific:
    - UPDATE STATISTICS
    - CREATE/DROP/ALTER PARTITION
      - CREATE PARTITION FUNCTION
      - CREATE PARTITION SCHEME
      - ALTER TABLE SWITCH PARTITION
      - ALTER PARTITION SCHEME [NEXT USED]
      - ALTER PARTITION FUNCTION [SPLIT RANGE | MERGE RANGE]
      - TRUNCATE TABLE
      - requires 2016: TRUNCATE TABLE [WITH(PARTITIONS(n))]
  - Partition Query:
    - SELECT sys.tables, sys.indexes, sys.partition_schemes, sys.partition_functions, sys.partitions, sys.partition_range_values

- **Oracle Tuning**:

© Copyright 2000-2017 TIBCO Software Inc.

- Increase the archiver timeout to 2 hours for each process.
- Increase the deletion of the archive logs to 1 hour if errors start occurring referencing rollback segments or archive space issues.
- Increase the UNDO tablespace to 50 GB or higher if errors start occurring regarding rollback segment too old.
- Calibrate the I/O (dbms_resource_manager.calibarate_io).

© Copyright 2000-2017 TIBCO Software Inc.

3 New Installation and Configuration

Supported Database Platforms

The majority of metrics discussed in the previous sections are generated using custom aggregation procedures. Because DV does not retain the system metrics data needed to generate KPImetrics data long enough for historical reporting, the KPImetrics module must store this cached data to a dedicated database in order to retain the generated results.

The KPImetrics module supports the following database platforms at this time as incremental caching targets.

- Oracle 11g, 12c or later (recommended due to better push-down capabilities)
- SQL Server 2016 or higher or SQL Server Azure
    - ***Previous implementations of KPImetrics using SQL Server 2012/2014 must migrate their database to SQL Server 2016.***

Support for additional platforms would require customization of the KPImetrics module by a professional services consultant. Please contact TIBCO's professional service group for details.

Please note that it is strongly recommended that the database chosen to cache KPImetrics data have case sensitivity and ignore trailing space settings that match your DV server to maximize query pushdowns in order to minimize the amount of additional load the KPImetrics module adds to your DV environment.

Installing KPImetrics

Turn off DV metrics

Turn off DV metrics if it is running.

1. Use DV Studio and open /policy/metrics under the root folder at the bottom of Studio
2. If metrics is "enabled", uncheck the box to turn if off.

Download and Import the KPImetrics components to your DV instance

Deploy the KPImetrics components to your DV instance in order to use the KPImetrics module.

3. **Download** the ASAssets Utilities and KPImetrics from the TIBCO Open Source GIT site
    a. Utilities: Utilities_2020Q200.zip
        i. Follow the Utilities documentation "How To Use Utilities.pdf" for installation.

© Copyright 2000-2017 TIBCO Software Inc.

Do this first.

ii. Import Utilities_2020Q200.car

iii. **IMPORTANT** – Configure the environment procedure uniquely for each DV environment: /shared/ASAssets/Utilities/environment/getEnvName

1. Example: DEV, TEST, PROD

b. KPImetrics: KPImetrics_2020Q202.zip

i. KPImetrics_YYYYQnnn_installation.car

ii. KPImetrics_YYYYQnnn.car

iii. KPImetrics Configuration Guide vx.y.pdf

iv. KPImetadata Configuration Guide vx.y.pdf

v. KPImetrics_Table_Relationship_Diagram.pptx

vi. KPImetrics Overview.pdf

vii. KPImetrics_worksheet.xlsx

viii. KPImetrics_scripts directory

If **upgrading** an existing KPImetrics installation

1. **Proceed to Section 4 "Upgrading KPImetrics"**

If **new installation**, complete the following steps to configure the KPImetrics components:

2. **Upload CAR file:**

   a. Upload KPImetrics_YYYYQnnn.zip to the target DV server file system and record the full path for use during installation.

3. **Login to DV Studio as "admin"**

   a. All configuration operations should be completed as DV "admin".

4. **Import KPImetrics Installation CAR file**:

   a. In the Studio left resource panel tree, right click on the root folder (/) icon and select Import.

   b. Import the file **KPImetrics_YYYYQnnn_installation.car** with the overwrite checkbox enabled. The folders /shared/ASAssets/KPImetrics_installation and should appear after the import completes.

5. **Execute Pre-Installation Script**:

   a. Execute "**1_Pre_Installation**" and provide parameters:

      i. Location: /shared/ASAssets/KPImetrics_installation/1_Pre_Installation

      ii. IN **metrics_app_id_password** - DV password for the user/owner of KPImetrics source code which is "metrics_app_id".

      iii. IN **car_file_os_full_path** - Full path to the car file archive in the OS file system. If null the import is skipped and the KPImetrics_YYYYQnnn.car will need to be imported manually.

© Copyright 2000-2017 TIBCO Software Inc.

b. Information Only Section

i. This script performs the following operations:

ii. Create the published data source "*ASAssets*" if it does not already exist.

iii. Create the "KPImetrics" catalog for ASAssets data source.

iv. Create a DV user called "metrics_app_id" in the "composite" domain. This way to can determine the process id that is executing requests and filter these requests out of the metrics history tables if you choose, using a strategy to be discussed later.

v. Import KPImetrics_YYYYQnnn.car if path is provided.

6. **Import KPImetrics CAR File**: [Optional if not done in step 4.]

a. **Bypass this step if you provided the path in Step 4**. above and the car file was successfully imported. Otherwise proceed with the instructions below.

b. In the Studio left resource panel tree, right click on the root folder (/) icon and select Import.

c. Import the file **KPImetrics_YYYYQnnn.car** with the overwrite checkbox enabled. The folders /shared/ASAssets/KPImetrics and /services/databases/ASAssets/KPImetrics appear after the import completes.

Configuration Overview

The following is an overview of resources to be configured prior to post-installation execution.

Recommended approach: Use **KPImetrics_worksheet.xlsx** as a way to gather the required information for this section.

1. Configure KPI_DV_AdminInterface Data Source
2. Configure KPImetrics Data Source
3. Configure DV Email
4. Configure Common Values Procedure
5. Configure KPIMetrics Job Lookup Tables
6. Configure KPImetrics Triggers
7. Deploy CPU and Memory Checker shell scripts (Windows or UNIX)
8. Configure LDAP
9. Configure Metadata

[1.] Configure the KPI_DV_AdminInterface data source

This data source is used as a loopback admin interface data source. The trigger

© Copyright 2000-2017 TIBCO Software Inc.

"kpimetricsTrig_34_DBMSRebuildIndexes → pRebuildIndexesAdminInterface" requires the ability to update metadata which can only be done as an admin user. Therefore, this provides a mechanism for the trigger to call out of DV through the admin interface data source which sets the user context thus allowing the metadata to be updated.

1. **Configure KPI_DV_AdminInterface data source**:
    a. Configure the port, user and password as needed.
    b. The host can remain localhost.
    c. Note: The admin password **"MUST BE"** the same on all nodes as this interface may be used on any given node since metadata is replicated in a cluster.
2. Note: This datasource will be updated dynamically by pRebuildIndex depending on the cluster node that is chosen to execute that procedure. Only the hostname and port will be modified. The procedure pRebuildIndexes runs once per cluster, once a week.

[2.] Configure the KPImetrics data source

The KPImetrics module makes use of several custom tables to store logging and metrics data. You must configure a data source connection in order to view KPImetrics data.

1. **Configure KPImetrics data source**:
    a. Locate and configure the appropriate data source for your KPImetrics database.
        i. NOTE: The recommended user is DVKPI but any user will work fine.
        ii. **Oracle**: **Recommended schema (user)=DVKPI**
            1. /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_11g
            2. /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_12c
        iii. **SQL Server**: **Recommended database name=DVKPI**
            1. /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016
    b. Enable the data source that is required.
    c. Test the connection to make sure it is working properly.
    d. Disable the data sources not being used.
    e. If you used a schema name or database name different than DVKPI, the post installation script will automatically take care of this.

[3.] Configuration Parameters

1. **Configure DV email: Used to send alerts and notifications**
    a. Select Administration → Configuration → Server → Configuration → E-mail
        i. From Address
        ii. SMTP Host Name
        iii. SMPT Port
        iv. If required, provide user name and password

© Copyright 2000-2017 TIBCO Software Inc.

b. Test sending an email using /lib/util/SendEMail
   i. Do not put anything in the “from” address. Leave blank.
   ii. Populate the following:
      “to”=<your email>
      1. “subject”=test
      2. “contentType”=TEXT_PLAIN
      3. “content”=test

2. **Configure Weight of Time Keeper**
   a. Select Administration → Configuration → Server → Configuration → Cluster → Cluster trigger distribution
      i. Weight of time keeper: 0
         1. Set this if time keeper needs special weight in weighted round robin trigger distribution. For most use cases this value need not be set.
         2. Set to 0 so that the timekeeper node does “not” participate in executing the triggers.
3. **Postgres Repository Pool Settings**
   a. Select Administratioin → Configuration → Server → Configuration → Repository Database → Connection Pools
      i. System Pool
         1. Pool Maximum Size (On Server Restart): 200
         2. Default=50
         3. Maximum number of internal database connections opened for use by the Server metadata sub-system. Note that this number must be greater than the "Minimum". This value is locally defined. It will not be altered when restoring a backup and will not be replicated in a cluster.
      ii. System Table Pool
         1. Pool Maximum Size (On Server Restart): 200
         2. Default=50
         3. Maximum number of internal database connections opened for client access to system tables. Note that this number must be greater than the "Minimum". This value is locally defined. It will not be altered when restoring a backup and will not be replicated in a cluster.
4. **For Microsoft SQL Server Data source target only**
   a. Select Administratioin → Configuration → Data Sources →MS SQLServer Sources
      i. <u>Microsoft BCP utility path</u>: The path on the TDV server where Microsoft BCP is installed.
         1. Default=<not set>
         2. The absolute file path of the Microsoft BCP utility. This value is

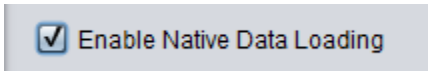
© Copyright 2000-2017 TIBCO Software Inc.

locally defined. It will not be altered when restoring a backup and will not be replicated in a cluster.

ii. Column Delimiter: |~|
    1. Default=<not set>
    2. The column delimiter character to be used for the BCP utility while loading data. If not defined, the value in Configuration > Data Sources > Common to Multiple Source Types > Data Sources Data Transfer > Column Delimiter will be used.
    3. If SQL Server and BCP is configured, use "|~|" for the "MS SQL Server Sources" "Column Delimiter" instead of just a "|" symbol. This is because the data will contain pipe symbols "|". The "|~|" makes the delimiter more unique where the data should not contain this pattern. Search for "Column Delimiter" in Administration→Configuration.

b. Select Administratioin → Configuration → Data Sources →Common to Multiple Source Types
    i. Data Sources Data Transfer → Buffer Flush Threshold. The minimum value is 1000. Change to 50000
    ii. Default Commit Row Limit: Change to 50000

c. Enable Native Loading on the datasource
    i. Open KPI_sqlserver_2016
        1. Click on the "Advanced" tab
        2. Enable Native Data Loading and save the datasource.



[4.] Configure Common Values

The KPImetrics module uses several constant values that are set in the procedure /shared/ASAssets/KPImetrics/Customize/**commonValues**. You will need to update some of these constants with values for your environment to ensure that KPImetrics functions correctly. Update the following minimum values for operation.

1. **Configure Common Values:**

    a. Open the procedure /shared/ASAssets/KPImetrics/Customize/commonValues and modify the following properties:

    d. **cisServerNickname** – Configure with the DV server nickname for the instance of DV. E.g. DEV1, SIT1, UAT1, PROD1. This is used in email notifications to alert you of issues. You determine what your environment nicknames are. Be consistent. *Each DV server environment must be unique*. For a DV cluster, each node in the cluster will have the same nickname as it describes the environment and not the node.

© Copyright 2000-2017 TIBCO Software Inc.

e. **Cluster Awareness Configuration:**
   i. The following variables are used to designate a dedicated time keeper host and port. None of the ClusterSafeCache procedures should be executed on a dedicated time keeper node because general processing and work is never routed to this node. It's only job in the cluster is to service triggers and perform deployments. However, rule #4 is the exception to this recommendation. Execute the following procedure on the "dedicated" time keeper node to get the hostname and port: /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pGetSystemInformation
   ii. Rules:
      1. SINGLE NODE: Set the hostname and port to null when this environment is a single node and not a cluster.
      2. CLUSTER NO TIMEKEEPER: Set the hostname and port to null when this environment is a cluster and any node in the cluster may be the timekeeper and all nodes do work.
      3. CLUSTER WITH TIMEKEEPER: Set the hostname and port to the the dedicated timekeeper node and port when the environment is a cluster and the node has been dedicated as a timekeeper and does not do work.
      4. CLUSTER WITH TIMEKEEPR CAPTURE: Set the hostname and port to null when the environment is a cluster and metadata should be captured for the timekeeper even though it is dedicated and does not perform work. Note: Using this option will require running the KPImetrics procedures on this node thus requiring it do work.
   iii. **dedicatedTimeKeeperHostname** – dediated time keeper host
   iv. **dedicatedTimeKeeperPort** – dedicated time keeper port.
   v. **totalNumberWorkerNodes** – designates the total number of worker nodes used for processing.
   vi. Rules:
      1. SINGLE NODE: Single node (stand-alone) environment. No cluster. The total should be the default of 1. Example: totalNumberWorkerNodes=1
      2. CLUSTER NO TIMEKEEPER: Cluster environment with NO "dedicated" timekeeper. The total is the total number of nodes in the cluster. Example: a 3-node cluster with NO dedicated timekeeper. totalNumberWorkerNodes=3
      3. CLUSTER WITH TIMEKEEPER: Cluster environment with a "dedicated" timekeeper. The total is the total number of nodes minus 1. Example: a 3-node cluster with a dedicated timekeeper. totalNumberWorkerNodes=2

© Copyright 2000-2017 TIBCO Software Inc.

4. CLUSTER WITH TIMEKEEPR CAPTURE: Cluster environment with a "dedicated" timekeeper where KPImetrics runs on the timekeeper node to capture metrics and metadata. The total is the total number of nodes in the cluster. Example: a 3-node cluster with a dedicated timekeeper. totalNumberWorkerNodes=3

f. **Purge History:**

i. This is used to purge various tables. These are the measured in days and should be aligned with the partitionNumber which is measured months. If partitionNumber=4 then the values below should be 120. The exception the above rule is purgeMetadata. Due to the incredible amount of data that it collects, it should be tuned down do more like 30 days or less.

ii. **purgeWorkflowData** – Purge tables: METRICS_CIS_WORKFLOW, METRICS_JOB_DETAILS, METRICS_EVENT_REG_LOG, METRICS_EVENT_REG_LOG_LINEAGE

iii. **purgeSQLRequests** – Purge tables: METRICS_SQL_RESOURCE_LINEAGE, METRICS_SQL_COLUMNS, METRICS_SQL_RESOURCE, METRICS_SQL_REQUEST, METRICS_SQL_CONTROL_LOG

iv. **purgeResourceUsage** – Purge tables: METRICS_CIS_SYSTEM_RESOURCES, METRICS_CPU_MEMORY_CHECKER, METRICS_LOG_DISK, METRICS_LOG_IO, METRICS_LOG_MEMORY, METRICS_SYS_DATASOURCES, METRICS_SYS_CACHES

v. **purgeHistory** – Purge tables: metrics_sessions_hist, metrics_requests_hist, metrics_resources_usage_hist when partitionNumber=0.

vi. **purgeMetadata** – RULE: purgeMetadata - Purge tables: All METADATA_% tables. A valid value must be between 1 and 366. Anything else will throw an exception. The Cache_METADATA_TABLES will always purge the current day prior to loading new data. The amount of data is always the current day + the history. To keep 3 days would be the current day - 2 days of history. Example: To keep 3 days of metadata the value will be 3. The other partitions will be truncated as per the formula: [*startDate*=CURRENT_TIMESTAMP *endDate*=DATEADD('day', (366-3), CURRENT_TIMESTAMP)]. The startDate and endDate are passed into the P_METADATA_TRUNCATE_PARTITION function. All partitions are truncated from the current day until -3 days from current day within the 366 interval partitioning scheme. Therefore there are 3 total days of data available to query. Rule executed by .../Metadata/System/ClusterSafeCache/Cache_METADATA_TABLES().

© Copyright 2000-2017 TIBCO Software Inc.

g. **replyTo** – Provide a single email as a reply to.
h. **sendTo** – Provide the comma separated list of email address to send notifications to. This setting is used as a fall back if there are no METRICS_EVENT_REGISTRATION rows configured and, in the event, that the database is completely down. Note: You must configure email in DV.
i. **CpuCheckerCommandPath** – Determine which script format to use based on your installation environement.
    i. The following section stores CPU and memory checker default values for WINDOWS and UNIX. The following are the command line execution statements:
    ii. CPU Utilization - processor time percentage
    iii. Windows 10: ' powershell.exe -file "$DV_HOME/bin/KPImetricsCpuUtilization.ps1"'
    iv. UNIX (Linux 6): '$DV_HOME/bin/KPImetricsTopCommandGrepCpu_linux6.sh'
    v. UNIX (Linux 7): '$DV_HOME/bin/KPImetricsTopCommandGrepCpu_linux7.sh'
j. **memoryCheckerCommandPath** – Determine which script format to use based on your installation environement.
    i. Memory Utilization - Memory used and memory available
    ii. Windows 10: 'powershell.exe -file "$DV_HOME/bin/KPImetricsMemUtilization.ps1"'
    iii. UNIX (Linux 6): '$DV_HOME/bin/KPImetricsFreeMemCommand_linux6.sh'
    iv. UNIX (Linux 7): '$DV_HOME/bin/KPImetricsFreeMemCommand_linux7.sh'
k. **dataSourceName** – Provide the name of the data source used to store KPImetrics data. Valid values are KPI_oracle_11g, KPI_oracle_12c, KPI_sqlserver_2016
l. **dataSourceCatalog** – Identifies the data source catalog name if applicable [SQL Server]. Set to null if not applicable.
m. **dataSourceSchema** – Identifies the data source schema name [Oracle, SQL Server].
n. **collectionTablespaceName** – Identifies the Metrics collection tablespace name which will contain the DV out-of-the-box metrics tables. To use the default tablespace or no tablespace set this value to NULL. You should have already created a tablespace (with no logging if Oracle) or a filegroup if SQL Server.
o. **historyTablespaceName** – Identifies the Metrics history tablespace name which will contain all of the history and reporting tables. To use the default tablespace or no tablespace set this value to NULL. You should have already created a tablespace or a filegroup if SQL Server.
p. **indexTablespaceName** – Identifies the Index tablespace name which will contain the indexes for the history tables. To use the default tablespace or no tablespace set this value to NULL. You should have already created a tablespace or a filegroup if SQL Server.
q. **historyTableCompression** – Identifies the type of compression that is allowed for the history tables [metrics_sessions_hist, metrics_resources_usage_hist, metrics_requests_hist]. Applied on table creation only.

© Copyright 2000-2017 TIBCO Software Inc.

i. Oracle:
    1. NOCOMPRESS - used for an Oracle database that does not support compression
    2. COMPRESS FOR QUERY HIGH - used for an Oracle Exadata platform.
    3. COMPRESS - basic compression.
ii. SQL Server: parameters is ignored and should be set to null

r. **partitionNumber** – Identifies the number of table partitions to use for the metrics history tables. The default is 0 indicating no partitioning will be used. It is “highly” recommended to partition the three history tables as it will make queries more performant and make it easier to drop a partition to purge data rather than executing delete statements. Recommendation: Always add 1 additional month to your standard data retention policy to allow for dropping a partition of data.
    i. (LLE) lower-level env: policy=3. Set to 4 months of data
    ii. (PROD) production-level env: policy=12. Set to 13 months of data.

s. **partitionStartDate** – Identifies the starting date for the partitioning scheme in the format 'YYYY-MM-DD'. If null then no partitioning date is used. If partitionNumber is > 0 and partitionStartDate is null an error is thrown. If the partition format is not correct and it cannot be cast to a DATE then an error is thrown. This should be the first of the month in which metrics are turned on and capturing data. If metrics have been previously turned on and have been capturing data for a while then use the first of the month of the date they were turned on. Execute the following query against the current metrics table to determine the date: select min(starttime) starttime from metrics_requests

t. **dataTransferBatchInsertMax** – Identifies the insert batch maximum number for the P_METRICS_ALL_TABLES data transfer procedure. This identifies the insert batch size commit when transferring data from the metrics_requests collection table into the metrics_requests_stg stage table. For SQL Server, this value governs inserts into all 3 staging tables. Note: This is currently implemented for Oracle and SQL Server and defaults to 50000.

u. **dataTransferBatchDeleteMax** – Identifies the delete batch maximum number for the P_METRICS_ALL_TABLES and P_METRICS_DELETE_COLLECTION data transfer procedures. This identifies the delete batch size commit when deleting collection table data for metrics_sessions, metrics_requests and metrics_resources_usage and the staging tables. Note: This is currently only implemented for SQL Server and defaults to 50000.

v. **dataTransferBatchUpdateMax** – Identifies the batch maximum number for the P_METRICS_ALL_TABLES data transfer procedure. After the metrics_requests collection table is transfered into the metrics_requests_hist table, an update routine executes to correlate the resourcekind and dataservicename from the metrics_resources_usage_hist with the

© Copyright 2000-2017 TIBCO Software Inc.

metrics_requests_hist table. This routine commits rows from the metrics_requests_stg table for the batch number specified here. Note: This is currently implemented for Oracle and SQL Server and defaults to 50000.

[5.] Configure Metrics Job Lookup Tables

The KPImetrics contains several lookup tables that need to be pre-populated prior to the installation script running. The tables include:

- METRICS_JOB_ENVIRONMENTS
- METRICS_JOB_FILTERS
- METRICS_EVENT_REGISTRATION

1. **Configure pqInsert_METRICS_JOB_tables**
   a. Edit /shared/ASAssets/KPImetrics/Customize/pqInsert_METRICS_JOB_tables
      i. Configure the ***METRICS_JOB_ENVIRONMENTS***
         1. Add a unique row for each environment in your pipeline. Remove any example rows that you are not relevant.
         2. Example: DEV1, SIT1, UAT, PROD
      ii. Configure the ***METRICS_JOB_FILTERS*** – The job filters are used to filter out rows by the data transfer routine when transferring from the collection tables to the history tables. The objective is to identify potentially high-use, non-user defined requests that are of little or no value to the ultimate KPImetrics reporting. *Since this feature is not built into DV metrics, it is "highly" recommended to filter out admin and other user accounts associated with deployment or non-query type of activity in DV*. Additionally, rows can be filtered out based on the resource path and resource type.
         1. Example 1 would be for each environment:
            ENV_TYPE TABLE_NAME USER DOMAIN RESOURCE_KIND
            'UAT', 'metrics_resources_usage', 'admin', 'composite', 'system'
         2. Example 2 would be any of the KPImetrics functionality that produces their own events. The assumption is that the KPImetrics folder resources were configured for resource ownership by metrics_app_id during installation:
            ENV_TYPE TABLE_NAME USER DOMAIN RESOURCE_KIND
            'UAT', 'metrics_resources_usage', 'metrics_app_id', 'composite', 'system'
         3. Example 3 would be a deployment process that produces an inordinately high volume of events. Assume you have a deployment

© Copyright 2000-2017 TIBCO Software Inc.

user called “deploy_app_id” that runs a deployment process but you don’t want to log those events:

```
ENV_TYPE TABLE_NAME              USER        DOMAIN RESOURCE_KIND
'UAT', 'metrics_resources_usage', 'deploy_app_id', 'composite', 'system'
```

4. Add a unique row for each environment, user, domain and resourcekind combination.
5. Example 4: Suppose you had a particular resource path and type that you wanted to filter out. Insert a row for each environment like this:

```
ENV_TYPE TABLE_NAME     USER DOMAIN RESOURCE_KIND
RESOURCE_PATH RESOURCE_TYPE
'UAT', 'metrics_resources_usage', null,null,null,
'/lib/resource/GetColumnDependencies', 'PROCEDURE'
```

b. Post-installation maintenance
   i. This script can be executed by itself post-installation by simply executing it to reload what is configured.

2. **Configure *pqInsert_METRICS_EVENT_REGISTRATION***

a. Edit /shared/ASAssets/KPImetrics/Customize/pqInsert_METRICS_EVENT_REGISTRATION
   a. **Requirement**: Prior to execution of this script LDAP data must first be configured and loaded. The post-installation script will automatically cache LDAP_PERSON as long as it is configured and caching is turned on server-wide.
   b. Configure the METRICS_EVENT_REGISTRATION
      1. This table contains the event registrations for sending emails based on the subscriptions to various events. This procedure is used for either initial load or maintenance. If a row already exists, it does not update it. It simply bypasses it. This means that you can run this procedure as many times as you want and not impact existing rows. It does not delete or unsubscribe requester events. To delete a subscription, invoke pMetricsEventRegistrationUnsubscribe(). A subscription in the METRICS_EVENT_REGISTRATION table consists of a unique record for the combination of SUBSCRIBER_EMAIL, GROUP_NAME,

© Copyright 2000-2017 TIBCO Software Inc.

ENVIRONMENT_TYPE, EVENT_TYPE and REQUESTER_EMAIL.

2. Use Case 1 – This provides a way to historically log Long Running Events and Memory Exceeded Events, amongst the other events.
3. Edit a row to provide the subscription information. At a minimum, add a group subscriber email for the DV administration group for each event. If you don't have a group alias, then choose the DV administrator's email to receive alerts. A row includes the following:
    a. REQUESTER_EMAIL [PK] – Primary requester email.
    b. SUBSCRIBER_EMAIL [PK] – userid email or a group email alias. Who the email alert will be sent to.
    c. GROUP_NAME [PK] – Group name subscribing to. When an alert occurs for a user the groups will be checked and cross-referenced with this registered group. The group [all] is a composite group and a catch-all for any user belonging to this composite group.
    d. ENVIRONMENT_TYPE [PK] – Register for all environments [ALL] or a certain environment type [DEV1, CIT1, SIT1, UAT, TT, PROD]
    e. EVENT_TYPE [PK] – [LONG_RUNNING, EXCEEDED_MEMORY, INACTIVITY, WORKFLOW_FAILURE, DBMS_SCHEDULER_ERROR, PURGE_HISTORY, METRICS_FAILURE]
    f. EXCLUDE_TEXT – A comma separate list of text that when found will signal exclusion and the email will not be sent. This is a way of filtering out emails based on text.

c. Post-installation maintenance
    i. This script can be executed by itself post-installation by simply executing it and answering "Y" to the parameter to delete the rows and reload what is configured.

[6.] Configure KPImetrics Triggers

The KPImetrics module uses a series of triggers to cache various tables of information. The different triggers provide flexibility to turn on and off processing as required. If certain functionality and data is not required, the trigger can be turned off saving on database space

© Copyright 2000-2017 TIBCO Software Inc.

and DV processing.

1. **Configure default triggers for your use case**

   a. Open/Edit the resource /shared/ASAssets/KPImetrics/Customize/**defaultTriggersToEnable**
   b. Only modify the ON/OFF settings for each trigger. Leave all other settings alone.
      i. Refer to the following sections for details on each trigger:
         1. Metadata System Triggers and Load Scripts
   c. The current triggers defaulted to OFF are as follows:
      i. **kpimetricsTrig_00_CheckMetricsActivityDebug** – Only turn this on if you suspect that DV metrics is not working properly and you want to debug the DV metrics every hour.
      ii. **kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST** – This trigger is defaulted to OFF. If you wish to perform SQL parsing on the request description SQL statement to parse out the table and column resources used in the SQL then turn this trigger on. There is quite a bit of overhead associated with this trigger. It will compete with regular user requests for CPU and memory. Consider tweaking the trigger to run after hours and before regular work hours sufh as 6pm to 8am.
      iii. **kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS** – Only turn this on if you get a code update from the Open Source site and there were changes to the SQL Parser code.
      iv. **kpimetricsTrig_17_CheckExceedMemoryPercentRequests** – You may choose to keep this off in lower-level environments but turn it on in PROD environments.
      v. **kpimetricsTrig_18_CheckLongRunningRequests** – You may choose to keep this off in lower-level environments but turn it on in PROD environments.
         1. **Administration** configuration is required for this to work.
         2. DV Server Attribute that sets the long running request limit in minutes. This is defined by the DV server attribute value found at '/server/event/generation/requests/requestRunMinutes'.
         3. Search for '**Request Run Time**' in Administration --> Configuration screen in DV Studio. This controls the period of time after which a request is considered to be long-running, resulting in the generation of a RequestRunForTooLong event, if request events are enabled. A value of 0 disables this feature.

© Copyright 2000-2017 TIBCO Software Inc.

4. Typical value is 60 minutes.

vi. **kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE** – This trigger is defaulted to OFF. If kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST is turned off then turn this trigger off as it uses data produced by that trigger to produce lineage data. It will compete with regular user requests for CPU and memory. Consider tweaking the trigger to run after hours and before regular work hours sufh as 6pm to 8am.

vii. **kpimetricsTrig_33_DeleteCollection [deprecated]** – This trigger must remain OFF.

1. 2020Q100: 2020-01-14: As of the 2020Q100 release this trigger "**MUST**" remain off. It is currently deprecated.
2. Collection table clean-up is managed by DV out-of-the-box metrics based on the /policy/metrics advanced section settings. Those settings should be set to [1 day] to keep metrics and [1 hour] to truncate rows.

viii. **kpimetricsTrig_35_DBMSDeleteCheck** – This trigger is used to monitor when TDV metrics executes the delete statements against the collection tables. It is optional but may provide value in terms of getting an idea what is being executed by TDV metrics.

ix. **kpimetricsTrig_40_Cache_METADATA_TABLES** – This trigger is defaulted to OFF. This trigger executes the KPI Metadata strategy. It is intensive which is why it only runs once a day. It could be set to run less often if metadata does not change that often. It should never run more than once a day as it will put load on the DV server. Refer to “KPImetadata Configuration Guide v1.3.pdf” for more information about configuration and usage. Before turning this on, the following procedure must be configured: /shared/ASAssets/KPImetrics/Customize/pqInsert_METADATA_Constants.

d. **Trigger Category – Essential**

i. Provides baseline data for the data transfer and other processes

1. kpimetricsTrig_01_Cache_ALL_RESOURCES
2. kpimetricsTrig_02_Cache_ALL_USERS
3. kpimetricsTrig_03_Cache_LDAP_PERSON

ii. Used to transfer data from collection tables into history tables

1. kpimetricsTrig_30_DBMSScheduler

iii. Used to check for errors with the data transfer

1. kpimetricsTrig_31_DBMSSchedulerError

© Copyright 2000-2017 TIBCO Software Inc.

iv. Used to perform partition management on the history tables
    1. kpimetricsTrig_32_DBMSPartitionManager
v. Provides clean-up/purge capability based on define schedule
    1. kpimetricsTrig_16_PurgeHistoryData
vi. Provides monitoring capabilities
    1. kpimetricsTrig_14_CheckCISWorkflowStatusFail
    2. kpimetricsTrig_15_CheckMetricsActivity
vii. Provides status table maintenance for METRICS_CIS_WORKFLOW and METRICS_JOB_DETAILS
    1. kpimetricsTrig_24_ValidateUpdateStatusTables
    2. kpimetricsTrig_25_ExecuteMetricsSqlControl
viii. Provides index rebuild capabilities
    1. kpimetricsTrig_34_DBMSRebuildIndexes [relies on kpimetricsTrig_24…]
ix. Metrics system event triggers – monitor health of metrics
    1. kpimetricsTrig_50_MetricsPersistenceFailure
    2. kpimetricsTrig_51_MetricsTruncateFailure
    3. kpimetricsTrig_52_MetricsBackupFailure
    4. kpimetricsTrig_53_MetricsRestoreFailure

e. **Trigger Category – 2nd level processing (lite-weight)**
    i. A series of value-added metrics that are an addition to the DV out-of-the-box metrics that are lite-weight in terms of processing. DV does not incur much overhead when these triggers execute.
        1. kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES
        2. kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER
        3. kpimetricsTrig_06_Cache_LOG_DISK
        4. kpimetricsTrig_07_Cache_LOG_IO
        5. kpimetricsTrig_08_Cache_LOG_MEMORY
        6. kpimetricsTrig_12_Cache_SYS_CACHES
        7. kpimetricsTrig_13_Cache_SYS_DATASOURCES
        8. kpimetricsTrig_17_CheckExceedMemoryPercentRequests
        9. kpimetricsTrig_18_CheckLongRunningRequests
        10. kpimetricsTrig_19_AllCustom_AccessByUserOverTime
        11. kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime
        12. kpimetricsTrig_21_AllCustom_ResourceCount_Details
        13. kpimetricsTrig_22_AllCustom_ResourceCount_Total
        14. kpimetricsTrig_35_DBMSDeleteCheck

© Copyright 2000-2017 TIBCO Software Inc.

f. **Trigger Category – 3rd level processing (heavy-weight)**
   i. A series of value-added metrics that are an addition to the DV out-of-the-box metrics
      1. kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST
      2. kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE
      3. kpimetricsTrig_40_Cache_METADATA_TABLES

[7.] Deploy CPU and Memory Checker shell scripts (Windows or UNIX)

Install CPU and Memory Checker scripts into the DV Server file system.

a. Install CPU and Memory Checker scripts into the $DV_HOME/bin directory on either Windows or UNIX depending on your operating system
b. Set the file permissions accordingly especially for UNIX such as [rwxr-xr-x]. The recommend location to deploy the scripts is $DV_HOME/bin so that they can be executed by the user account that DV is running under.
c. Take note of where the scripts have been deployed, you will need to provide the path to the scripts when configuring the KPImetrics "commonValues" script. The data source "/shared/ASAssets/KPImetrics/Physical/Metadata/CPUAndMemChecker" is used to execute the scripts. The following details the scripts:
   a. Windows Powershell
      i. KPImetricsCpuUtilization.ps1
      ii. KPImetricsMemUtilization.ps1
   b. Linux6_scripts
      i. KPImetricsFreeMemCommand_linux6.sh
      ii. KPImetricTopCommandGrepCpu_linux6.sh
   c. Linux7_scripts
      i. KPImetricsCpuFormat_linux7
      ii. KPImetricsFreeMemCommand_linux7.sh
      iii. KPImetricTopCommandGrepCpu_linux7.sh

[8.] Configure LDAP

The KPImetrics module is designed to retrieve user data from an LDAP directory server in order to provide additional detail on which users are making use of a monitored DV environment. You may configure the LDAP data source provided with the KPImetrics module to connect to your corporate LDAP directory server. The essence of many of the queries is based on LDAP data and LDAP users who connect to DV to execute queries. If you do not have LDAP, then consider the options below.

A note about debugging. Never query the LDAP1/LDAP2 datasource tables directly. You may encounter an error. Always use LDAP_PERSON to query. Experience has shown that a regularly formatted query is necessary to execute against most LDAP systems. They generally

© Copyright 2000-2017 TIBCO Software Inc.

don't allow SELECT * FROM <ldap_table>.

1. **Configure the LDAP data source**
   a. **Option 1** – Configure the KPImetrics LDAP data source to connect to your local ldap: /shared/ASAssets/KPImetrics/Physical/Metadata/LDAP1.
      i. Proceed to #2 below to map LDAP_PERSON to the LDAP data source.
      ii. NOTE: If more than one DV domain exists, then configure LDAP2. Add additional LDAP data sources for each domain. For the LDAP_PERSON view, a UNION statement will be needed between each SELECT for each LDAP data source domain that is represented.
   b. **Option 2** – Configure your own LDAP data source.
      i. Proceed to #2 below to map LDAP_PERSON to your data source.
      ii. NOTE: If more than one DV domain exists, then configure LDAP2. Add additional LDAP data sources for each domain. For LDAP_PERSON view, a UNION statement will be needed between each SELECT for each LDAP data source domain that is represented.
   c. **Option 3** – Configure a pseudo LDAP to satisfy the inner workings of KPImetrics.
      i. Proceed directly to #2 below and modify the existing UNION statement with composite users. There is no mapping to LDAP in this scenario.
2. **Modify the LDAP_PERSON view /shared/ASAssets/KPImetrics/Customize/LDAP_PERSON**
   Correctly map to your LDAP directory structure.

- For Option 1 and 2, comment out the entire default UNION section of 'user1', 'user2' and 'user3' at the bottom of the view.
  ii. Uncomment the top section which is the LDAP implementation section.
  iii. For multiple LDAP data source "domains", you will need a SELECT statement for each one with a UNION to tie them altogether.
  iv. Comment out the bottom section containing the "UNION" selects.
  v. Determine which table the LDAP data should come from. The default is "user" but it may be person, organizationPerson, ingetOrgPerson or a custom table.
  vi. [**required**] Modify the source location "FROM" clause as required to point to the correct table that was determined in the previous step:
     1. FROM /shared/ASAssets/KPImetrics/Physical/Metadata/LDAP1/"user"
  vii. [**required**] Create the proper transformation for "**userkey**" so that the data can be joined with metrics history "userkey" data. Analyze the LDAP data and make sure the "userkey" will contain the proper data that will join with metrics history "userkey" which is derived from metrics_resources_usage_hist."user".
  viii. [**required**] Create the proper 'value' for "**domainkey**" so that the data can be joined with metrics history "domainkey" data. The "domainkey"

© Copyright 2000-2017 TIBCO Software Inc.

will be the name of the DV domain which can be found in the DV Manager browser under Domain Configuration. There is a direct correlation between the LDAP data source connection and the domain connection. The names need to be “exactly” the same in order to properly join with metrics history “domainkey” which is derived from metrics_resources_usage_hist.”domain”.

- For Option 3, modify the existing UNION and modify ‘user1’, ‘user2’, and ‘user3’ with your own composite users.
- Note: “userkey” is required.
- Note: “domainkey” is required.
- Guidelines:
    ix. Keep the alias column names the same.
    x. The physical column names may be different based on your LDAP data source.
    xi. If a column does not exist then simply do a CAST(null as VARCHAR(…)) columnName.
    xii. Keep the number of columns the same.
    xiii. Keep the datatype and length the same.
    xiv. Implement a where clause as required based on your LDAP data source. E.g. WHERE userid IS NOT NULL AND employeeStatus IS NOT NULL and employeeStatus = ‘A’ – active employees only
- Key fields to map include the following:
    xv. cn (lower case cn field)
    xvi. uid
    xvii. userkey (lower case cn|uid|employeeID)
        1. Transformation of the user key which could be cn or uid or employeeID. Use SQL CASE if needed.
        2. This field is required and is used to join with metrics history tables. The data needs to be normalized so that it matches with metrics history “userkey” data [metrics_resources_usage_hist.userkey].
    xviii. CAST(CASE WHEN mail IS NOT NULL AND INSTR(mail, '@') > 0
    THEN SUBSTRING(mail, INSTR(mail, '@')+1)
    ELSE mail
    END AS VARCHAR(255)) domainkey,
        1. Transformation of mail to get the domain name being used.
    xix. objectClass
    xx. displayName
    xxi. name
    xxii. objectGUID (user id field)
    xxiii. sn (surname)
    xxiv. givenName

© Copyright 2000-2017 TIBCO Software Inc.

xxv. employeeNumber
xxvi. employeeID
xxvii. mail
xxviii. baseDN
xxix. relativeDN

- Nice to have fields to map include the following:
  xxx. description
  xxxi. telephoneNumber
  xxxii. c (country)
  xxxiii. l (city)
  xxxiv. st (state)
  xxxv. street (street address)
  xxxvi. ou (organization unit)
  xxxvii. title
  xxxviii. postalAddress
  xxxix. postalCode (zip code)
  xl. postOfficeBox
  xli. initials
  xlii. employeeType
  xliii. manager
  xliv. homePhone
  xlv. mobile

[9.] Configure Metadata

KPImetrics Metadata is a new feature that captures metadata from the Data Virtualization Server. There are four constants tables that require configuration data. This data is inserted based on the configuration of the procedure /shared/ASAssets/KPImetrics/Customize/pqInsert_METADATA_Constants.

Refer to "**KPImetadata Configuration Guide v1.x.pdf**" for details on configuration of metadata.

Once completed decide whether this feature should be turned on by setting the triggerAction from OFF to ON for "kpimetricsTrig_40_Cache_METADATA_TABLES" in the following procedure: /KPImetrics/Customize/defaultTriggersToEnable.

Execute Post-Installation Script

The KPImetrics module provides an automated script to complete the installation. The "Information Only Section" below will describe in detail what the script is going to execute. When the installation is complete and there are no red/impacted resources then turn on the triggers to begin processing KPI metrics data.

© Copyright 2000-2017 TIBCO Software Inc.

Execute Installation Script

Perform the post-installation configuration.

1. **Execute Post-Installation Script**:
    a. WARNING: If you are executing an installation into an existing KPImetrics environment, this script **WILL DROP** existing tables and data. Do not use this script for upgrading an environment. We recommend using TIBCO Professional Services for doing upgrades.
    b. Execute "**2_Post_Installation**" and provide parameters:
        i. Location: /shared/ASAssets/KPImetrics_installation/2_Post_Installation
        ii. IN inDebugSecondary – Debug Secondary provides a deep debug with detailed debug output. Y=deep debug. N=cursory, high-level debug. N is recommended unless there are issues.
        iii. IN **performInstallationAction** – Y=perform the installation which will drop and recreate KPImetrics tables/sequences/procedures. N=Do nothing.
        iv. IN **destroyCIS_metrics_collection_Tables** – Y=destroy and recreate DV metrics collection tables including metrics_requests, metrics_resources_usage, and metrics_sessions. N=do not destroy if they exist.
        v. IN **forceOverwrite** – Force overwriting the CPU and Memory checker Windows/UNIX scripts. Y=force overwrite. N=do not force overwrite.
        vi. IN **userTransformation** – Leave null if no transformation is required. Contains a native database SQL-based transformation that is valid within the context of SELECT and uses "user" as the column name to transform. It must be a valid transformation for Oracle or SQL Server. If null, the default is simply "user". The column name that is "user" is transformed into the "userkey". The same transformation is applied to metrics_sessions_hist, metrics_requests_hist and metrics_resources_usage_hist. The objective is for the "userkey" field data to match the METRICS_LDAP_PERSON."userkey" field. This will only be applicable when the actual "user" requires a transformation to match with data found in LDAP.
            1. Always use double quotes around the "user" field.
            2. Never include an alias in front of the "user" field. Only reference the "user" field itself within the transformation.
            3. The user transformation is actually applied on the fly during the creation of the data transfer script for either Oracle or SQL Server.

© Copyright 2000-2017 TIBCO Software Inc.

When these scripts are instantiated in the database, they will contain the transformation which will take the “user” field and normalize it into the “userkey” field:

a. 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_[700|800]

b. 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_[700|800]

4. Example of an extreme case.

   a. Given a metrics_resources_usage_hist.“user” = u12345678.

   b. Given METRICS_LDAP_PERSON.”userkey” = 12345678 where the u is dropped or assumed.

   c. Input Oracle Transformation would be:

```
CASE WHEN INSTR('abcdefghijklmnopqrstuvwxyz',
SUBSTR(LOWER("user"),1,1)) > 0 and INSTR('0123456789',
SUBSTR("user",2,1)) > 0

THEN SUBSTR(LOWER("user"),2)

ELSE LOWER("user")

END
```

   d. Input SQL Server Transformation would be:

```
CASE WHEN patindex(SUBSTRING(LOWER("user"),1,1),
'abcdefghijklmnopqrstuvwxyz') > 0 and patindex(SUBSTRING("user",2,1),
'0123456789') > 0

THEN SUBSTRING(LOWER("user"),2,len("user"))

ELSE LOWER("user")

END
```

vii. IN **domainTransformation** – Leave null if no transformation is required. Contains a native database SQL-based transformation that is valid within the context of SELECT and uses "domain" as the column name to transform. It must be a valid transformation for Oracle or SQL Server. If null, the default is simply "domain". The column name that is "domain" is transformed into the "domainkey". The same transformation is applied to metrics_sessions_hist, metrics_requests_hist and metrics_resources_usage_hist. The objective is for the "domainkey" field data to match the METRICS_LDAP_PERSON."domainkey" field. This will only be applicable when the actual "domain" requires a transformation to match with data found in LDAP.

   1. Always use double quotes around the “domain” field.

   2. Never include an alias in front of the “domain” field. Only reference the “domain” field itself within the transformation.

   3. The domain transformation is actually applied on the fly during the

© Copyright 2000-2017 TIBCO Software Inc.

creation of the data transfer script for either Oracle or SQL Server. When these scripts are instantiated in the database, they will contain the transformation which will take the “domain” field and normalize it into the “domainkey” field:

a. 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script
b. 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script

4. Example of an extreme case.
   a. Most likely a transformation will not be needed unless the DV domain does not match the LDAP email domain. If that is the case, then a CASE statement will be required to map the LDAP email domain to the DV domain.
   b. Given a metrics_resources_usage_hist.“domain” = “tib1”.
   c. Given METRICS_LDAP_PERSON.”domainkey” = “tibco1.com”.
   d. Input Oracle Transformation would be:

```
CASE WHEN LOWER("domain") = "tib1" THEN LOWER("tibco1.com")
WHEN LOWER("domain") = "tib2" THEN LOWER("tibco2.com")
ELSE LOWER("domain")
END
```

   e. Input SQL Server Transformation would be:

```
CASE WHEN LOWER("domain") = "tib1" THEN LOWER("tibco1.com")
WHEN LOWER("domain") = "tib2" THEN LOWER("tibco2.com")
ELSE LOWER("domain")
END
```

Deploy CPU and Memory Checker shell scripts (Windows and UNIX)

IF ERROR ONLY

During post-installation, it was attempted to write the scripts to the $DV_HOME/bin directory on either Windows or UNIX depending on your operating system.

d. If no error was returned then all scripts were installed correctly into $DV_HOME/bin and are able to be executed. Continue to the next major step/section.
e. If the error message “*23. INSTALL KPImetrics CpuAndMemoryChecker scripts manually.*” was received during post-installation, then you will need to install the scripts manually and set the file permissions accordingly especially for UNIX such as [rwxr-xr-x]. The recommend location to deploy the scripts is $DV_HOME/bin so that they can be executed by the user account that DV is running under. Take note of where the

© Copyright 2000-2017 TIBCO Software Inc.

scripts have been deployed, you will need to provide the path to the scripts when configuring the KPImetrics “commonValues” script. The data source “/shared/ASAssets/KPImetrics/Physical/Metadata/CPUAndMemChecker” is used to execute the scripts. The following details the scripts:

a. Windows Powershell
   i. KPImetricsCpuUtilization.ps1
   ii. KPImetricsMemUtilization.ps1
b. Linux6_scripts – **be sure to set permissions**
   i. KPImetricsFreeMemCommand_linux6.sh
   ii. KPImetricTopCommandGrepCpu_linux6.sh
c. Linux7_scripts – **be sure to set permissions**
   i. KPImetricsCpuFormat_linux7
   ii. KPImetricsFreeMemCommand_linux7.sh
   iii. KPImetricTopCommandGrepCpu_linux7.sh

Enable Triggers

Enabling triggers starts the processing of KPI metrics data. The KPImetrics module makes use of pure database tables in order to retain DV metrics for a longer period than supported by the base DV logging functionality.

Please note that executing this procedure will cache the minimum required tables: METRICS_ALL_RESOURCES, METRICS_ALL_USERS and METRICS_LDAP_PERSON. Those 3 tables must be completed before any of the other triggers are allowed to run. The following concept is now being used:

1. **Concept**:
   a. Cache_LDAP_PERSON is executed immediately to load the latest LDAP information.
   b. A control record is added METRICS_SQL_CONTROL for Cache_ALL_RESOURCES and Cache_ALL_USERS for each node in the cluster.
      i. A subsequent process wakes up on the trigger “kpimetricsTrig_25_ExecuteMetricsSqlControl” to execute these for each node.
      ii. Depending on the timing of this trigger there are other dependent procedures that may also execute the one that they need based on this rule:
         1. Cache_ALL_RESOURCES gets executed by dependents:
            a. Cache_ALL_RESOURCES
            b. Cache_METRICS_SQL_REQUEST_EXEC
            c. Cache_METRICS_SQL_REQUEST_EXEC_ADHOC
            d. Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS
            e. Cache_METRICS_SQL_RESOURCE_LINEAGE
            f. pMETRICS_ALL_TABLES_exec

© Copyright 2000-2017 TIBCO Software Inc.

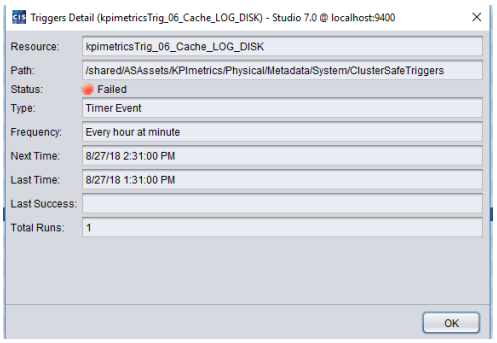
2. Cache_ALL_USERS gets executed by dependents:
    a. Cache_ALL_USERS
    b. pMETRICS_ALL_TABLES_exec

2. **<u>Enable Triggers</u>**:
    a. Execute the **updateTriggers** procedure
        i. Location: /shared/ASAssets/KPImetrics/Configuration/**updateTriggers**
        ii. *<u>enable</u>* – 0=disable, 1=enable, 2=display trigger list
            1. Select 1 to enable all configured triggers.
        iii. *<u>includeList</u>* – Comma-separated list of trigger numbers to include in the (enable/disable) action. Leave null if the "defaultTriggersToEnable" are configured as desired.
        iv. *<u>excludeList</u>* – Comma-separated list of trigger numbers to exclude in the (enable/disable) action. The excludeList overrides includeList. Leave null if the "defaultTriggersToEnable" are configured as desired.
    b. Review the Studio Manager → Triggers tab and look for any "Failed" triggers
        i. Keep checking this screen until you have verified that at least one of the triggers has executed successfully. If the trigger has fired there will be a timestamp in the "Last Time" column. If the trigger is red "Failed" then go then proceed to point ii. below. If the trigger is green "Ready" then it is executing without issues. The majority of the triggers will fire within an hour of turning them on.

kpimetricsTrig_00_CheckMetricsActivityDebug	Disabled
kpimetricsTrig_01_Cache_ALL_RESOURCES	Ready
kpimetricsTrig_02_Cache_ALL_USERS	Ready
kpimetricsTrig_03_Cache_LDAP_PERSON	Ready
kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES	Ready
kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER	Ready
kpimetricsTrig_06_Cache_LOG_DISK	Failed
kpimetricsTrig_07_Cache_LOG_IO	Ready
kpimetricsTrig_08_Cache_LOG_MEMORY	Ready
kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST	Ready
kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCE...	Disabled
kpimetricsTrig_12_Cache_SYS_CACHES	Ready
kpimetricsTrig_13_Cache_SYS_DATASOURCES	Ready
kpimetricsTrig_14_CheckCISWorkflowStatusFail	Failed
kpimetricsTrig_15_CheckMetricsActivity	Ready
kpimetricsTrig_16_PurgeHistoryData	Ready
kpimetricsTrig_17_CheckExceedMemoryPercentRequests	Disabled
kpimetricsTrig_18_CheckLongRunningRequests	Disabled
kpimetricsTrig_19_AllCustom_AccessByUserOverTime	Ready
kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime	Ready
kpimetricsTrig_21_AllCustom_ResourceCount_Details	Ready
kpimetricsTrig_22_AllCustom_ResourceCount_Total	Ready
kpimetricsTrig_30_DBMSScheduler	Ready
kpimetricsTrig_31_DBMSSchedulerError	Failed
kpimetricsTrig_32_DBMSPartitionManager	Ready

        ii. If a trigger has failed
            1. Unfortunately, double clicking on the trigger does not yield any additional insights as shown in the screen shot

© Copyright 2000-2017 TIBCO Software Inc.



2. Trigger failures will most likely show up in the "**cs_server_events.log**". It also may be necessary to check "**cs_server.log**".

iii. The most common solution to a trigger failure will be permission problems on resources.

1. Make sure the privileges are set correctly on /shared/ASAssets/Utilities folder. It is recommended that /shared/ASAssets/Utilities be set to the composite group "all" with Read Execute Select.
2. Specifically, check privileges for /shared/ASAssets/Utilities/environment/getEnvName().

iv. To verify the above issue and recommendation, check for the following in the "**cs_server_events.log**". There would be an entry in that log that looks similar to this:

2018-08-27 13:31:00.002 -0400 INFO START trigger name=/shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_06_Cache_LOG_DISK type=TIMER action=PROCEDURE 1 metrics_app_id composite 20400 6141564240009 6141564240009 /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_06_Cache_LOG_DISK TIMER PROCEDURE

2018-08-27 13:31:00.112 -0400 ERROR FAIL trigger name=/shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_06_Cache_LOG_DISK type=TIMER action=PROCEDURE, reason failed: **User "metrics_app_id/composite" has insufficient privileges to execute "/shared/ASAssets/Utilities/environment/getEnvName",** on line 55, column 65. User has READ privileges for that resource. [repository-1900321] 1 metrics_app_id composite 20402 6141564240449 6141564240009 /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_06_Cache_LOG_DISK TIMER PROCEDURE User

© Copyright 2000-2017 TIBCO Software Inc.

"metrics_app_id/composite" has insufficient privileges to execute "/shared/ASAssets/Utilities/environment/getEnvName", on line 55, column 65. User has READ privileges for that resource. [repository-1900321]

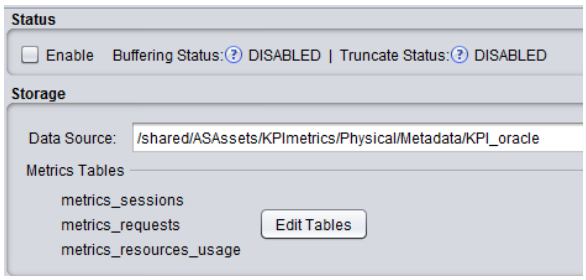
v. Verify the trigger issue is resolved. The screen shot indicates that the previously failed trigger now has successfully executed identified by a timestamp in the "Last Success" field and a status of green "Ready".

Resource	Status	Type	Frequency	Next Time	Last Time	Last Success	Total Runs
kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST	Ready	Timer Event	Every 30 minutes	8/27/18 2:45:00 PM	8/27/18 2:15:00 PM	8/27/18 2:15:00 PM	2

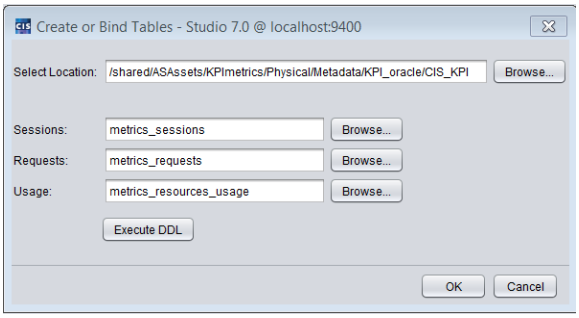
Configure DV Out-of-the-box Metrics

To enable DV metrics:

1. Locate and open /policy/metrics
2. Since the tables were created earlier, choose the option to configure the path to the tables without re-creating them.
3. Data Source – browse and set the data source to the database you have configured.
4. Select "Edit Tables"



5. Browse to the schema path
   a. Browse to the Sessions table: metrics_sessions
   b. Browse to the Requests table: metrics_requests
   c. Browse to the Usage table: metrics_resources_usage
   d. DO NOT execute DDL
   e. Click OK to finish

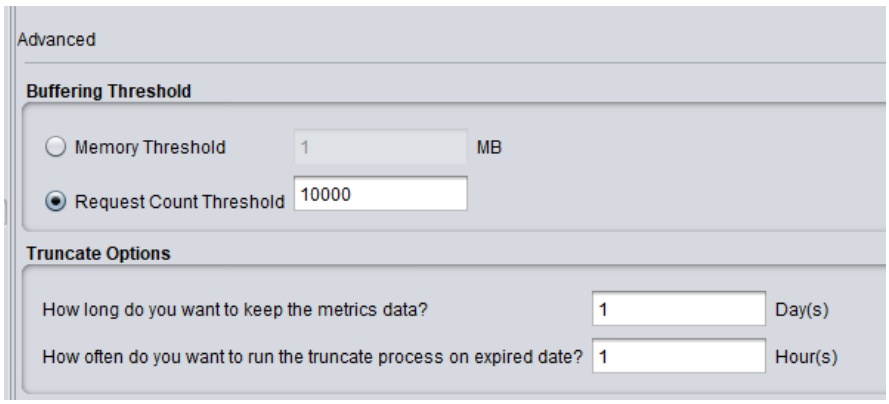


© Copyright 2000-2017 TIBCO Software Inc.

a. Configure the Advanced Section

i. As of Release 2020Q100 it is "**REQUIRED**" to set the following:

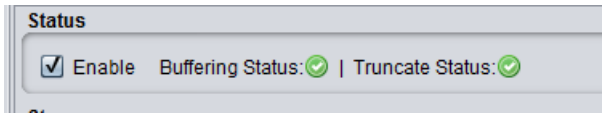
- *Request Count Threshold*: Set to **10000**. Adjust accordingly for active servers. It may need a higher number so that the number of inserts is reduced.
- *How long do you want to keep the metrics data?*: **1 Day(s)**
  1. KPImetrics is no long deleting collection data. It will rely on the DV server to execute delete commands to clean-up the collection tables. The clock starts ticking when metrics is enabled. A setting of 1 day means that there will be a minimum of 24 hours of data that will stay in the collection tables. This is a rolling 24 hours.
- *How often do you want to run truncate process on expired data?*: **1 Hour(s)**
  1. It is a good idea to delete once an hour to keep the collection tables as small as possible. DV metrics will issue the delete command once an hour based on when DV metrics was enabled. This is what keeps the data at a rolling 24 hours when the above setting is set to "1 Day".
- The KPImetrics "kpimetricsTrig_30_DBMSScheduler" trigger executes every 2 hours to process the collection data since the last execution and insert it into the history tables. It tracks min (last) values based on METRICS_JOB_DETAILS values. Always processes given a range of id's and timestamps.
- Important: If you turn off the KPImetrics triggers but leave metrics running, you will lose transactions because the data is no longer being processed into the history tables.



b. Enable metrics and save.

i. Make sure both radio button indicators are green.

6. Enable metrics and save



a. DV metrics will issue **delete** statements when **enabling** metrics:

- Delete statements are qualified by nodehost, nodeport and logintime for metrics_sessions or starttime for metrics_requests and metrics_resources_usage.

© Copyright 2000-2017 TIBCO Software Inc.

delete from /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_sessions where nodehost='localhost' and nodeport=9800 and logintime < '2020-01-20 17:27:08.756'

delete from /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_requests where nodehost='localhost' and nodeport=9800 and starttime < '2020-01-20 17:27:08.756'

delete from /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_resources_usage where nodehost='localhost' and nodeport=9800 and starttime < '2020-01-20 17:27:19.975'

b. DV metrics will issue **insert** statements when **disabling** metrics to flush the buffer to the collection tables.

insert into /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_sessions select * from /system/datasources/system/SYS_METRICS_SESSIONS

insert into /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_requests select * from /system/datasources/system/SYS_METRICS_REQUESTS

insert into /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_requests select * from /system/datasources/system/SYS_METRICS_REQUESTS

7. If the Buffer Status shows RED then review the following:
   a. Make sure the three collection tables exist in the database and metrics are properly configured to point to them.
   b. Make sure the tables have the correct permissions for writing to them.
   c. If Oracle, make sure the tablespace has the correct permissions for writing to it.
   d. If SQL Server and BCP is configured, use “|~|” for the “MS SQL Server Sources” “Column Delimiter” instead of just a “|” symbol. This is because the data will contain pipe symbols “|”. The “|~|” makes the delimiter more unique where the data should not contain this pattern. Search for “Column Delimiter” in Administration→Configuration.

KPImetrics Configuration is COMPLETE!

© Copyright 2000-2017 TIBCO Software Inc.

4 Upgrading KPImetrics

Introduction

This section describes how to upgrade KPImetrics from a previous version to the current version without data loss. The upgrade supports upgrading on the same TDV verision such as 7.x or upgrading the KPImetrics from TDV 7.x to 8.x. When upgrading from 7.x to 8.x, the collection tables will change. Specifically, metrics_requests and metrics_sessions add “user”, “domain” and “group” columns.

How to Upgrade KPImetrics

Start with a few house-keeping activities described below such as turning off metrics and determining your current version.

Requirements:

1. The upgrade script must be executed by the user admin or someone who has admin privileges.
2. Collection and staging tables will be dropped and recreated for both 7.x and 8.x upgrades.
3. Any version less than 2018.103 is not currently supported for upgrade. Contact Tibco Data Virtualization PSG if this use case comes up.

Migrating from 7.x to 8.x:

1. Upgrading will happen automatically in version 8.x
2. Simply export the KPImetrics from 7.x, import into 8.x and follow the upgrade steps below.
3. Collection and Staging tables will be dropped and recreated.

Current Version:

2. Turn off /***policy/metrics*** so no data is inserted into the metrics collection tables.
3. Turn off all triggers using
    a. ***/shared/ASAssets/KPImetrics/Configuration/updateTriggers(0)***
4. Clear out the collection tables:
    a. For any **KPImetrics *version previous to 2019Q402***, execute the following script manually *on each node of the cluster* in order to clear the collection tables:
        i. /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec
    b. For version ***2019Q402 and above***, execute the following script *once on one node* which will clear out the collection table for all nodes:
        i. /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec
    c. Execute this on 7.x prior to exporting for 8.x upgrade

© Copyright 2000-2017 TIBCO Software Inc.

   d. Note: collection and staging tables "WILL" be dropped and recreated during upgrade.

5. If migrating from 7.x to 8.x
   a. Turn off ***/policy/metrics*** on 8.x
   b. Export KPImetrics from 7.x using Studio 7.x
      i. /services/databases/ASAssets/KPImetrics
      ii. /services/databases/ASAssets/KPImetricsAdmin
      iii. /shared/ASAssets/KPImetrics
   c. Using Studio 8.x, Right-click on either Desktop (admin) or root hostname (/) and select "Import"
      i. Click on the "Overwrite" option. Leave all other options as they are.
      ii. Browse to the local file system for the car file specified in step a. above.
      iii. Click on preview and review what will be imported.
      iv. Click on Import and then click OK
   d. Configure KPI_DV_AdminInterface datasource
      i. /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_DV_AdminInterface
      ii. Configure hostname, port, login and password.
      iii. Test the datasource to make sure it successfully connects.
   e. If using the "SAME" database as 7.x then continue to step 5, this is an upgrade.
   f. If using a "DIFFERENT" database, then this is "NOT" an upgrade, it is a "NEW" installation. Follow the steps for a new installation in the manual.
6. If upgrading versions of TDV and creating a new TDV environment, ***create the user "metrics_app_id"*** on each node of the new environment.
   a. This applies to either a new 7.x or 8.x environment.
   b. This must be done prior to import of the KPImetrics car file.
7. The admin must take an ***export car file backup*** of the following folders:
   a. This step is not necessary if migrating from 7.x to 8.x as the exported car file serves as the backup.
      /services/databases/ASAssets/KPImetrics
      /services/databases/ASAssets/KPImetricsAdmin
      /shared/ASAssets/KPImetrics
8. Import KPImetrics Installation *** INSTALLATION .CAR ONLY ***
   a. File: ***KPImetrics_YYYYQnnn_installation.car***
      i. Example: KPImetrics_2019Q402_installation.car
   b. Using Studio, Right-click on either Desktop (admin) or root hostname (/) and select "Import"
      i. Click on the "Overwrite" option. Leave all other options as they are.
      ii. Browse to the local file system for the car file specified in step a. above.
      iii. Click on preview and review what will be imported.
      iv. If 8.x no .car file password is required as the .car file originiated in 7.x.
      v. Click on Import and then click OK
9. Verify if upgrade is allowed "BEFORE" import
   a. Execute ***/shared/ASAssets/KPImetrics_installation/validateUpgrade***
   b. If NOT - admin must contact Tibco PSG for a follow-on engagement.
   c. If YES, then the admin proceeds with the upgrade.
10. Backup existing customized resources

© Copyright 2000-2017 TIBCO Software Inc.

a. Execute ***/shared/ASAssets/KPImetrics_installation/backupCustomResources***.
b. The resources identified below are backed up.
c. Folder created: /shared/ASAssets/KPImetrics_backup/v_YYYY_MM_DD_YYYYQnnn

Key resources to be copied from /Metadata folder:

- /KPI_DV_AdminInterface data source if exists
- The active datasource starting with one of [KPI_oracle..., KPI_sqlserver...]
- The LDAP datasources which may be LDAP, LDAP1 or LDAP2

Key resources to be copied from /Customize folder:

- commonValues
- defaultTriggersToEnable
- LDAP_PERSON
- pqInsert_METADATA_Constants [formerly 07_pqInsert_KPI_Tables_METRICS_JOB_tables]
- pqInsert_METRICS_EVENT_REGISTRATION [formerly 08_pqInsert_KPI_Tables_METRICS_EVENT_REGISTRATION]
- pqInsert_METRICS_JOB_tables [formerly 08_pqInsert_KPI_Tables_METRICS_EVENT_REGISTRATION]

Key trigger metadata to be copied from /Metadata/System/ClusterSafeTriggers folder:

- All trigger metadata is captured and save into the backup folder as ../ClusterSafeTriggers/ClusterSafeTriggersView

11. Import the latest KPImetrics car file

****** STOP. READ. ******

a. DO NOT RUN THE CONFIGURATION "2_Post_Installation" WHICH DROPS AND RECREATES TABLES OR DATA WILL BE LOST.
b. File: ***KPImetrics_YYYYQnnn.car*** - e.g. KPImetrics_2019Q402.car
c. Using Studio, Right-click on either Desktop (admin) or root hostname (/) and select "Import"
   i. Click on the "Overwrite" option. Leave all other options as they are.
   ii. Browse to the local file system for the car file specified in step a. above.
   iii. Click on preview and review what will be imported.
   iv. Click on Import and then click OK
d. Import options checked:

X Caching	**X Data Source Connections**	Include Users Jars
X Include Cache Policies	**X Overwrite**	Merge Users Folder
Create Caching Tables	Override Locks	**X Privileges**
X Custom Jars	Discovery Files	

12. Execute "***3_Post_Upgrade***"

This script is used to upgrade an existing installation and bring its resources up to the most current version.

[Required] - The location of the backup target folder.
Example: /shared/ASAssets/KPImetrics_backup/v_2019_12_13_2019Q100
IN backupTargetFolder VARCHAR(4000),

[Required] - Cluster Awareness variable.

Designates the total number of worker nodes used for processing. This will be used to create working tables for each node.

Scenario 1 - stand-alone environment. No cluster. The total is 1.

© Copyright 2000-2017 TIBCO Software Inc.

Example: totalNumberWorkerNodes=1

Scenario 2 - cluster environment with NO "dedicated" timekeeper. The total is the total number of nodes in the cluster.

Example: a 3-node cluster with NO dedicated timekeeper. totalNumberWorkerNodes=3

Scenario 3 - cluster environment with a "dedicated" timekeeper. The total is the total number of nodes minus 1.

Example: a 3-node cluster with a dedicated timekeeper. totalNumberWorkerNodes=2

IN totalNumberWorkerNodes **INTEGER**,

[Required] - Cluster Awareness variables.
The following variables are used to designate a dedicated time keeper host and port.
None of the ClusterSafeCache procedures should be executed on a dedicated time keeper node because general processing and work is never routed to this node. It's only job in the cluster is to service triggers and perform deployments. Execute the following procedure on the "dedicated" time keeper node to get the hostname and port:
/shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pGetSystemInformation

Rules:

1. They both should be set to null when this environment is not a cluster (a.k.a. single node).
2. They both should be set to null when this environment is a cluster and any node in the cluster may be the timekeeper and all nodes do work.
3. Set the hostname and port to the the dedicated timekeeper node when the environment is a cluster and the node has been dedicated as a timekeeper and does not do work.

IN dedicatedTimeKeeperHostname **VARCHAR**,

IN dedicatedTimeKeeperPort **INTEGER**,

[Optional] - This is typically left null unless repeating an upgrade and you need to specify which steps to execute.
A numerical, comma-separated list of actions to execute as it relates to #9 in the notes above. If null, all actions are executed.

IN executeActionList **VARCHAR**,

[Optional] - This is typically left null unless a higher level of debug is needed.
Debug Secondary provides a deep debug with detailed debug output.
Y=deep debug. N=cursory, high-level debug. N is recommended unless there are issues.

IN indebugSecondary **CHAR(1),**

[Optional] - This is typically left null unless the circumstances specified in this description require it.
Contains a SQL-based transformation that is valid within the context of SELECT and uses "user" as the column name to transform. It must be a valid transformation for Oracle or SQL Server.
If null, the default is simply "user". The column name that is "user" is transformed into the "userkey".
The same transformation is applied to metrics_sessions_hist, metrics_requests_hist and metrics_resources_usage_hist.
The objective is for the "userkey" field data to match the METRICS_LDAP_PERSON."userkey" field. This will only be applicable when the actual "user" requires a transformation to match with data found in LDAP.

IN userTransformation **LONGVARCHAR,**

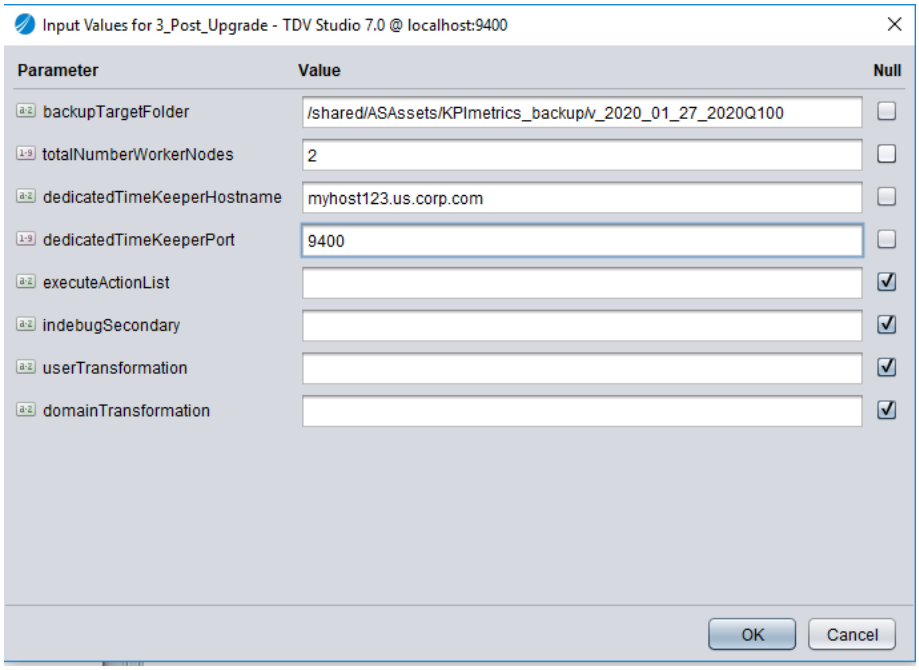
[Optional] - This is typically left null unless the circumstances specified in this description require it.
Contains a SQL-based transformation that is valid within the context of SELECT and uses "domain" as the column name to transform. It must be a valid transformation for Oracle or SQL Server.
If null, the default is simply "domain". The column name that is "domain" is transformed into the "domainkey".
The same transformation is applied to metrics_sessions_hist, metrics_requests_hist and metrics_resources_usage_hist.
The objective is for the "domainkey" field data to match the METRICS_LDAP_PERSON."domainkey" field. This will only be applicable when the actual "domain" requires a transformation to match with data found in LDAP.

IN domainTransformation **LONGVARCHAR,**

Example Screen shot:

© Copyright 2000-2017 TIBCO Software Inc.

Typically, the last 4 parameters are null.



Parameters – Additional Detail Explanation

userTransformation

1. Most likely a transformation will not be needed unless the DV user does not match the LDAP user. If that is the case, then a CASE statement will be required to map the LDAP user to the DV user via the userkey.

2. Always use double quotes around the “domain” field.

3. Never include an alias in front of the “domain” field. Only reference the “domain” field itself within the transformation.

4. The domain transformation is actually applied on the fly during the creation of the data transfer script for either Oracle or SQL Server. When these scripts are instantiated in the database, they will contain the transformation which will take the “domain” field and normalize it into the “domainkey” field: a. Given a metrics_resources_usage_hist.“user” = u12345678.

Example of an extreme case:

5. Given a metrics_resources_usage_hist.“user” = u12345678.5.

6. Given METRICS_LDAP_PERSON.”userkey” = 12345678 where the u is dropped or assumed.

7. Input Oracle Transformation would be:

```
CASE WHEN INSTR('abcdefghijklmnopqrstuvwxyz', SUBSTR(LOWER("user"),1,1)) > 0 and INSTR('0123456789',
SUBSTR("user",2,1)) > 0
THEN SUBSTR(LOWER("user"),2)
ELSE LOWER("user")
END
```

8. Input SQL Server Transformation would be:

```
CASE WHEN patindex(SUBSTRING(LOWER("user"),1,1), 'abcdefghijklmnopqrstuvwxyz') > 0 and
patindex(SUBSTRING("user",2,1), '0123456789') > 0
THEN SUBSTRING(LOWER("user"),2,len("user"))
ELSE LOWER("user")
```

© Copyright 2000-2017 TIBCO Software Inc.

```
END
```

domainTransformation

1. Most likely a transformation will not be needed unless the DV domain does not match the LDAP email domain. If that is the case, then a CASE statement will be required to map the LDAP email domain to the DV domain via the domainkey.

2. Always use double quotes around the “domain” field.

3. Never include an alias in front of the “domain” field. Only reference the “domain” field itself within the transformation.

4. The domain transformation is actually applied on the fly during the creation of the data transfer script for either Oracle or SQL Server. When these scripts are instantiated in the database, they will contain the transformation which will take the “domain” field and normalize it into the “domainkey” field:

Example of an extreme case:

5. Given a metrics_resources_usage_hist.“domain” = “tib1”.

6. Given METRICS_LDAP_PERSON.”domainkey” = “tibco1.com”.

7. Input Oracle Transformation would be:

```
CASE WHEN LOWER("domain") = "tib1" THEN LOWER("tibco1.com")
WHEN LOWER("domain") = "tib2" THEN LOWER("tibco2.com")
ELSE LOWER("domain")
END
```

8. Input SQL Server Transformation would be:

```
CASE WHEN LOWER("domain") = "tib1" THEN LOWER("tibco1.com")
WHEN LOWER("domain") = "tib2" THEN LOWER("tibco2.com")
ELSE LOWER("domain")
END
```

AUTOMATED steps to be performed [***Information Section Only***]:

a. Validate backed up copy version from getKPIVersion with the version from getKPIInstallationVersion.
    i. Don't proceed if return code = 0
b. Copy customized resources back to KPImetrics folder and enable datasources
    i. Copy resources
    ii. if exist /CustomizeOld,
        1. copy /shared/ASAssets/KPImetrics_backup/v_YYYY_MM_DD/CustomizeOld to /shared/ASAssets/KPImetrics/Customize
            a. commonValues,
            b. defaultTriggersToEnable,
            c. LDAP_PERSON
            d. pqInsert_METADATA_Constants,
            e. pqInsert_METRICS_EVENT_REGISTRATION,
            f. pqInsert_METRICS_JOB_tables
    iii. else
        1. Copy /shared/ASAssets/KPImetrics_backup/v_YYYY_MM_DD/Customize to /shared/ASAssets/KPImetrics/Customize
            a. commonValues,
            b. defaultTriggersToEnable,
            c. LDAP_PERSON
            d. pqInsert_METADATA_Constants,
            e. pqInsert_METRICS_EVENT_REGISTRATION,

© Copyright 2000-2017 TIBCO Software Inc.

f. pqInsert_METRICS_JOB_tables
iv. Copy /shared/ASAssets/KPImetrics_backup/v_YYYY_MM_DD/Metadata to /shared/ASAssets/KPImetrics/Physical/Metadata
v. Copy /shared/ASAssets/KPImetrics_backup/v_YYYY_MM_DD/ClusterSafeTriggers to /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers

c. Enable all copied datasources
d. Modify Common Values
   i. /shared/ASAssets/KPImetrics_installation/Helpers/modify_commonValues
e. Get the data source configuration parameters from commonValues
f. Modify Default Triggers
   i. /shared/ASAssets/KPImetrics_installation/Helpers/modify_defaultTriggersToEnable
g. Modify LDAP_PERSON
   i. /shared/ASAssets/KPImetrics_installation/Helpers/modify_LDAP_PERSON
h. Modify pqInsert_METRICS_JOB_tables
   i. /shared/ASAssets/KPImetrics_installation/Helpers/modify_pqInsert_METRICS_JOB_tables
i. Modify pqInsert_METADATA_Constants
   i. /shared/ASAssets/KPImetrics_installation/Helpers/Upgrade/modify_pqInsert_METADATA_Constants – Add EXCLUSION_LIST
   ii. /shared/ASAssets/KPImetrics_installation/Helpers/Upgrade/modify_pqInsert_METADATA_Constants_assign_privs – Add ASSIGN_PRIVILEGES
j. Modify pqInsert_METRICS_EVENT_REGISTRATION
   i. /shared/ASAssets/KPImetrics_installation/Helpers/Upgrade/modify_pqInsert_METRICS_EVENT_REGISTRATION – Add METRICS_FAILURE event type.
k. Rebind physical external packaged procedures to the datasource schema
   i. /shared/ASAssets/KPImetrics_installation/Helpers/rebindPhysicalDatabaseType('A')
l. Rebind the physical abstraction folder "/Physical/Abstraction" to the correct physical datasource folder "/Physical/KPI_[oracle|postgres|sqlserver]"
   i. /shared/ASAssets/KPImetrics_installation/Helpers/rebindPhysicalAbstraction
m. Drop the indexes
n. Execute the upgrade scripts based on the original version
   i. Read /shared/ASAssets/KPImetrics_backup/v_YYYY_MM_DD/getKPIVersion
      1. Pad version with 0's on the end 2019.3 = 2019.300
      2. Change the . to Q so that 2019.300 = 20193Q00
   ii. Get Database version for the DDL folder
   iii. Execute any scripts in /shared/ASAssets/KPImetrics_installation/DDL/UpgradeResources
      1. upgrade_2019Q401_to_2019Q402_04_InsertJobTables
   iv. Loop through the scripts in /shared/ASAssets/KPImetrics_installation/DDL/[Oracle|SqlServer]/Upgrade until 'upgrade_YYYYQnnn' matches the original version
   v. IF FOUND then execute the script
o. Introspect the KPImetrics datasource
p. Rebind physical database type tables to the datasource schema
   i. /shared/ASAssets/KPImetrics_installation/Helpers/rebindPhysicalDatabaseType('A')
q. Update impacted resources
r. Set resource privileges
s. Change resource ownership
t. Load the METRICS_JOBS table
u. Load the METRICS_EVENT_REGISTRATION table

© Copyright 2000-2017 TIBCO Software Inc.

v. Test the KPI_DV_AdminInterface data source
w. Remove DVKPI folder if not needed
x. Create the indexes
y. Installation and Configuration is complete.
    i. Configure the DV Out-Of-The-Box Metrics located at /policy/metrics
    ii. Execute: /shared/ASAssets/KPImetrics/Configuration/updateTriggers(1)

13. Configure KPI_DV_AdminInterface
    a. Make sure that this datasource is pointing back to the same server and the host, port, user and password are correct.
    b. Location: /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_DV_AdminInterface

14. Review Triggers
    a. The following trigger schedules have been overridden from the original settings:
        i. kpimetricsTrig_16_PurgeHistoryData – once a week SAT 2:30 AM
        ii. kpimetricsTrig_34_DBMSRebuildIndexes – once a week SUN 3:00 AM
        iii. kpimetricsTrig_01_Cache_ALL_RESOURCES – every 2 hours at 1:00 AM
        iv. kpimetricsTrig_02_Cache_ALL_USERS – every 2 hours at 1:00 AM
        v. kpimetricsTrig_30_DBMSScheduler – every 2 hours at 1:30 AM
            1. This should always be an odd hour on the half hour because it is depedent on kpimetricsTrig_01_Cache_ALL_RESOURCES which is every 2 hours on the top of the odd hour [1 am]. Therefore, this would follow 30 minutes after the latest METRICS_ALL_RESOURCE cache completion.
        vi. kpimetricsTrig_31_DBMSSchedulerError – every 2 hours a 2:00 am
            1. This should always be 30 min after kpimetricsTrig_30_DBMSScheduler
        vii. kpimetricsTrig_40_Cache_METADATA_TABLES – once a day at 11:30 PM
            1. This should always be an odd hour on the half hour because it is depedent on kpimetricsTrig_01_Cache_ALL_RESOURCES which is every 2 hours on the top of the odd hour [1 am]. Therefore, this would follow 30 minutes after the latest METRICS_ALL_RESOURCE cache completion.

15. Test each of the following scripts manually and in order shown to determine that there were no issues.
    a. After each script, look at the view to the right of the script to determine if is was successful or failed.
    b. If the corresponding trigger is turned off then there is no need to test the script.
    c. Scripts are located here: /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache

© Copyright 2000-2017 TIBCO Software Inc.

d. Validation views are located here:
   i. /shared/ASAssets/KPImetrics/Application/workflow/vCISWorkflowStatus
   ii. /shared/ASAssets/KPImetrics/Application/workflow/vJobDetails

SCRIPT	Validation View
-- Must do these three first to establish base line resources used by other scripts.	
1. Cache_ALL_RESOURCES	vCISWorkflowStatus
2. Cache_ALL_USERS	vCISWorkflowStatus
3. Cache_LDAP_PERSON	vCISWorkflowStatus
-- Quick tests	
4. Cache_CPU_MEMORY_CHECKER	vCISWorkflowStatus
5. Cache_LOG_DISK	vCISWorkflowStatus
6. Cache_LOG_IO	vCISWorkflowStatus
7. Cache_LOG_MEMORY	vCISWorkflowStatus
8. Cache_CIS_SYSTEM_RESOURCES	vCISWorkflowStatus
9. Cache_SYS_CACHES	vCISWorkflowStatus
10. Cache_SYS_DATASOURCES	vCISWorkflowStatus
11. pCheckExceedMemoryPercentRequests	vCISWorkflowStatus
12. pCheckLongRunningRequests	vCISWorkflowStatus
13. pPurgeData	vCISWorkflowStatus
-- Execute some external test queries to generate the proper requests.	
-- To force /policy/metrics to write data to the collection tables, disable it and then enable it.	
-- These scripts rely on data from collection tables	
14. pMETRICS_ALL_TABLES_exec	vJobDetails
15. pMETRICS_DELETE_COLLECTION_exec	vJobDetails
16. pPARTITION_MANAGER_exec	vJobDetails
17. pCheckDBMSSchedulerError	vJobDetails
18. Cache_AllCustom_AccessByUserOverTime	vCISWorkflowStatus
19. Cache_AllCustom_ActiveResourcesOverPeriodOfTime	vCISWorkflowStatus
20. Cache_AllCustom_ResourceCount_Details	vCISWorkflowStatus
21. Cache_AllCustom_ResourceCount_Total	vCISWorkflowStatus
22. Cache_METRICS_SQL_REQUEST_EXEC	vCISWorkflowStatus
23. Cache_METRICS_SQL_RESOURCE_LINEAGE	vCISWorkflowStatus
24. pCheckCISWorkflowStatusFail	vCISWorkflowStatus
-- Metadata Tests [**Only if configured**]	
25. Cache_METADATA_TABLES	vCISWorkflowStatus

16. Turn on ***/policy/metrics*** to start inserting rows into the metrics collection tables.

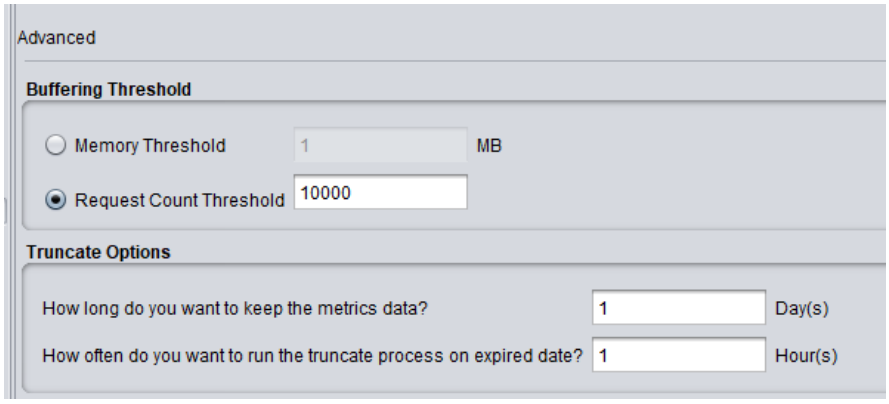
© Copyright 2000-2017 TIBCO Software Inc.

a. If the prior version used KPI_oracle or KPI_sqlserver then you will have to reconfigure it to use one of [KPI_oracle_11g, KPI_oracle_12c, KPI_sqlserver_2016]

b. Configure the Advanced Section

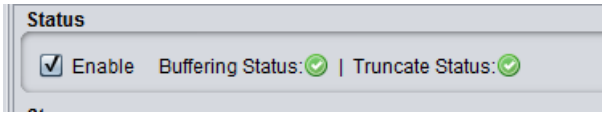
   i. As of Release 2020Q100 it is "**REQUIRED**" to set the following:

   - *Request Count Threshold*: Set to **10000**. Adjust accordingly for active servers. It may need a higher number so that the number of inserts is reduced.
   - *How long do you want to keep the metrics data?*: **1 Day(s)**
     1. KPImetrics is no long deleting collection data. It will rely on the DV server to execute delete commands to clean-up the collection tables. The clock starts ticking when metrics is enabled. A setting of 1 day means that there will be a minimum of 24 hours of data that will stay in the collection tables. This is a rolling 24 hours.
   - *How often do you want to run truncate process on expired data?*: **1 Hour(s)**
     1. It is a good idea to delete once an hour to keep the collection tables as small as possible. DV metrics will issue the delete command once an hour based on when DV metrics was enabled. This is what keeps the data at a rolling 24 hours when the above setting is set to "1 Day".
   - The KPImetrics "kpimetricsTrig_30_DBMSScheduler" trigger executes every 2 hours to process the collection data since the last execution and insert it into the history tables. It tracks min (last) values based on METRICS_JOB_DETAILS values. Always processes given a range of id's and timestamps.
   - Important: If you turn off the KPImetrics triggers but leave metrics running, you will lose transactions because the data is no longer being processed into the history tables.



c. Enable metrics and save.

   i. Make sure both radio button indicators are green.



   ii. DV metrics will issue **delete** statements when **enabling** metrics:
   - Delete statements are qualified by nodehost, nodeport and logintime for metrics_sessions or starttime for metrics_requests and

© Copyright 2000-2017 TIBCO Software Inc.

metrics_resources_usage.

delete from /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_sessions where nodehost='localhost' and nodeport=9800 and logintime < '2020-01-20 17:27:08.756'

delete from /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_requests where nodehost='localhost' and nodeport=9800 and starttime < '2020-01-20 17:27:08.756'

delete from /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_resources_usage where nodehost='localhost' and nodeport=9800 and starttime < '2020-01-20 17:27:19.975'

iii. DV metrics will issue **insert** statements when **disabling** metrics to flush the buffer to the collection tables.

insert into /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_sessions select * from /system/datasources/system/SYS_METRICS_SESSIONS

insert into /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_requests select * from /system/datasources/system/SYS_METRICS_REQUESTS

insert into /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_2016/DVKPI/dvkpi/metrics_requests select * from /system/datasources/system/SYS_METRICS_REQUESTS

d. If the Buffer Status shows RED then review the following:
   i. Make sure the three collection tables exist in the database and metrics are properly configured to point to them.
   ii. Make sure the tables have the correct permissions for writing to them.
   iii. If the problem persists, then check cs_server.log for errors.
   iv. If Oracle, make sure the tablespace has the correct permissions for writing to it.
   v. If SQL Server and BCP is configured, use “|~|” for the “MS SQL Server Sources” “Column Delimiter” instead of just a “|” symbol. This is because the data will contain pipe symbols “|”. The “|~|” makes the delimiter more unique where the data should not contain this pattern. Search for “Column Delimiter” in Administration→Configuration.

© Copyright 2000-2017 TIBCO Software Inc.

17. Turn on all triggers using
    /shared/ASAssets/KPImetrics/Configuration/updateTriggers(1)

18. Upgrade is complete.

© Copyright 2000-2017 TIBCO Software Inc.

5 Migrate a New KPImetrics Installation

The following instructions will demonstrate how to migrate a currently installed KPImetrics configuration from one TDV server environment to another. The new installation in the target environment does not have any data collected. If any data has been collected, it will get wiped out during this process.

The reason this is advantageous over a completely new install is that the effort of configuring most of the required resources has already been done. With a few minor tweaks after importing into the target environment, KPImetrics will be up and running in no time.

Perform the following steps in the source environment from which you will migrate from.

Export Source Environment .car file

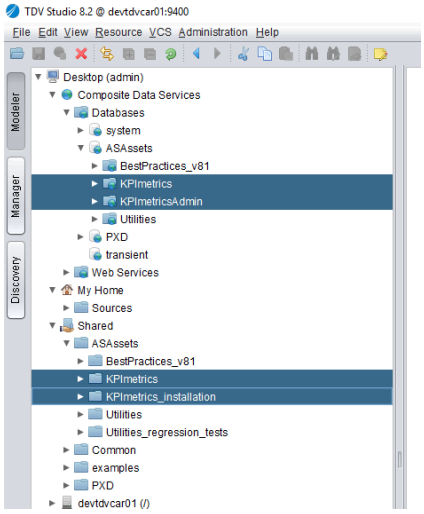
1. Sign into the TDV ***source*** environment with Studio as a user with administrator access.
2. Turn off the KPImetrics triggers using the following script:
    a. /shared/ASAssets/KPImetrics/Configuration/**updateTriggers(0)**

3. Export KPImetrics shared and database into a single .car file.
    a. Click on one folder and then Control-Click on the remaining to select all
        i. /services/databases/KPImetrics
        ii. /services/databases/ASAssets/KPImetricsAdmin
        iii. /shared/ASAssets/KPImetrics

© Copyright 2000-2017 TIBCO Software Inc.

iv. /shared/ASAssets/KPImetrics_installation

b. The screen shot below shows what this will look like:



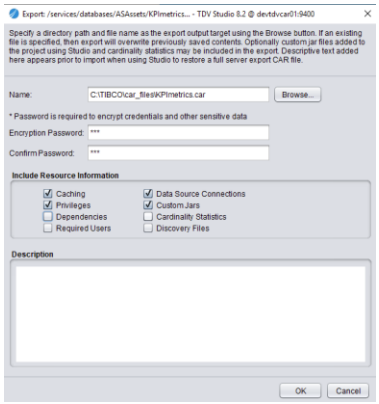
a. Right-click on the highlight KPImetrics and select “Export”.

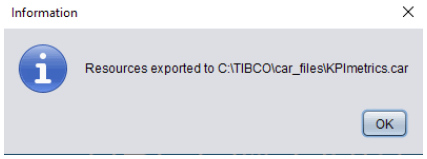
b. <u>Uncheck</u> “Dependencies” but leave the other boxes checked.

c. Navigate to a path to create the .car file and give it a name such as KPImetrics_<env>_<yyyymmdd>.car so that you know which environment it came from and what date it was exported.

d. Provide an encryption password.

e. Click OK to export.

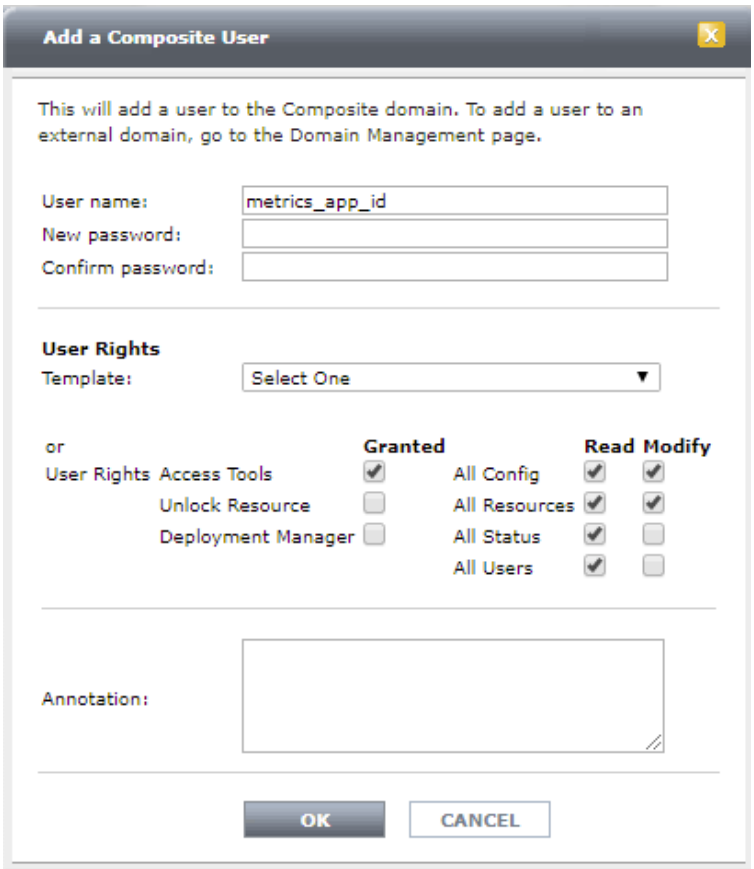




© Copyright 2000-2017 TIBCO Software Inc.

Import Target Environment .car file

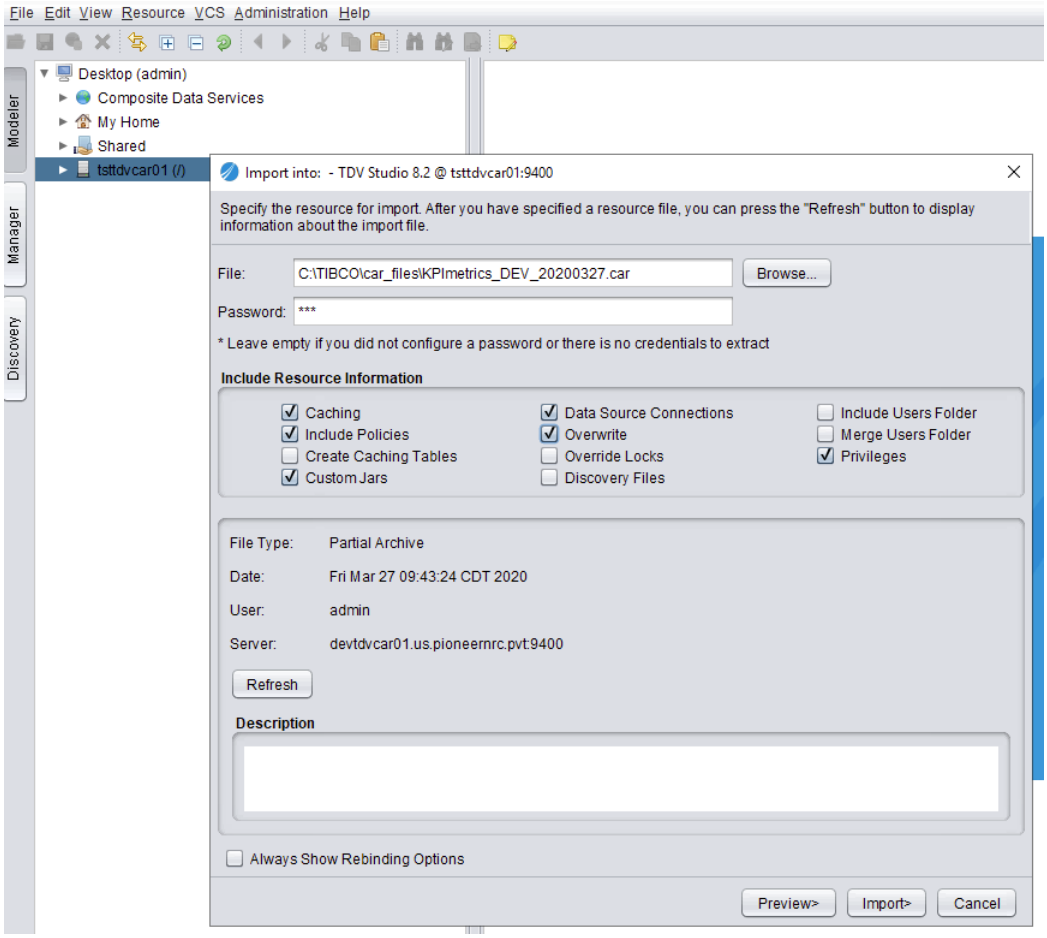
1. Sign into the TDV ***target*** environment with Studio as a user with administrator access.
2. Create the composite user “metrics_app_id” using TDV Manager
    a. http://<hostname>:9400/manager
    b. Select Security → User Management
    c. Select “All Domains” in the tab drop down
    d. Verify that the “metrics_app_id” user exists.
    e. If it does not exist then “Add User”
        i. Provide a password
        ii. Provide Studio Rights as shown in the screen shot below:



        iii. Click OK to create the user

© Copyright 2000-2017 TIBCO Software Inc.

3. Create the published database “ASAssets” if it does not exist.
    a. Note: It should already exist because the Utilities require it and the Utilities are a baseline for KPImetrics.
    b. If it is not there, then you will need to create it and reimport the Utilities – Read the Utilities instructions as it clearly states what to do.
4. Import KPImetrics_<env>_<yyyymmdd>.car file into the target environment
    a. Right-click on the root <hostname> (/) and select import.
    b. All check boxes should remain checked
    c. In addition click on “Overwrite”
    d. Enter the encryption password used during export
    e. See screen shot below:



© Copyright 2000-2017 TIBCO Software Inc.

f. Click on “Preview” to view what will be imported.

g. Click on “Import” to import the resources.

h. Click “OK” when it is finished.

Configure KPImetrics Target Installation

1. Sign into the TDV ***target*** environment with Studio as a user with administrator access.
2. Configure ***commonValues***
    a. Location: /shared/ASAssets/KPImetrics/Customize/commonValues
    b. Modify “**cisServerNickname**” for this environment
    c. Modify Cluster Awareness Configuration:
        i. The following variables are used to designate a dedicated time keeper host and port. None of the ClusterSafeCache procedures should be executed on a dedicated time keeper node because general processing and work is never routed to this node. It's only job in the cluster is to service triggers and perform deployments. Execute the following procedure on the "dedicated" time keeper node to get the hostname and port: /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pGetSystemInformation
        ii. The “Rules” are as follows:
            1. They both should be set to null when this environment is not a cluster (a.k.a. single node).
            2. They both should be set to null when this environment is a cluster and. any node in the cluster may be the timekeeper and all nodes do work.
            3. Set the hostname and port to the the dedicated timekeeper node when the environment is a cluster and the node has been dedicated as a timekeeper and does not do work.
        iii. **dedicatedTimeKeeperHostname** – dediated time keeper host
        iv. **dedicatedTimeKeeperPort** – dedicated time keeper port.
        v. **totalNumberWorkerNodes** – designates the total number of worker nodes used for processing.
            1. Scenario 1 - stand-alone environment. No cluster. The total should be the default of 1.
                a. Example: totalNumberWorkerNodes=1
            2. Scenario 2 - cluster environment with NO "dedicated" timekeeper. The total is the total number of nodes in the cluster.

© Copyright 2000-2017 TIBCO Software Inc.

b. Example: a 3-node cluster with NO dedicated timekeeper. totalNumberWorkerNodes=3

3. Scenario 3 - cluster environment with a "dedicated" timekeeper. The total is the total number of nodes minus 1.

c. Example: a 3-node cluster with a dedicated timekeeper. totalNumberWorkerNodes=2

d. Determine how long the data for this environment will be retained. Example: For LLE it is typically 4 months / 120 days. For PROD it is typically 13 months / 390 days.

i. **purgeWorkflowData**: LLE=120, PROD=390

ii. **purgeSQLRequests**: LLE=120, PROD=390

iii. **purgeResourceUsage**: LLE=120, PROD=390

iv. **purgeHistory**: LLE=120, PROD=390

v. **purgeMetadata**: LLE=30, PROD=30 – Due to the amount of data, the values are a lot lower for this setting.

vi. **partitionNumber**: LLE=4, PROD=13

e. Modify "**partitionStartDate**" – this is always the first day of the month in which you are doing the configuration.

f. If the catalog or schema is different than the environment you imported from then modify the following:

i. **dataSourceCatalog** – null if Oracle. Requires catalog for SQL Server. [Case Sensitive]

ii. **dataSourceSchema** – requires schema name [Case Sensitive]

g. If the tablespaces/filegroups are different in this database environment than the environment you imported from then modify the following. Leave null to use the primary tablespace or filegroup.

i. collectionTablespaceName

ii. historyTablespaceName

iii. indexTablespaceName

3. Install ***CPU_MEMORY_CHECKER*** scripts

© Copyright 2000-2017 TIBCO Software Inc.

a. Copy KPImetrics*.* scripts from the source TDV_HOME/bin folder to the target environment TDV_HOME/bin folder

b. For Linux, set the permissions properly

i. chmod 700 KPImetrics*

ii. chown <owner> KPImetrics*

iii. chgrp <group> KPImetrics*

c. vi KPImetricsTopCommandGrepCpu_linux6 or 7.sh

i. Modify CIS_HOME to point to the correct directory.

4. Modify the "**KPI_DV_AdminInterface**" data source

a. It should always point to *localhost* – do not change that.

b. Modify the port if necessary

c. Modify the user and password if necessary

d. Test the connection

5. Modify the database data source

a. For Oracle modify one of KPI_oracle_[11g|12c]

b. For SQL Server modify one of KPI_sqlserver_[2016]

c. Modify to point to the correct database host and port

d. Modify catalog and schema if necessary

e. Modify user and password if necessary

f. Test the connection

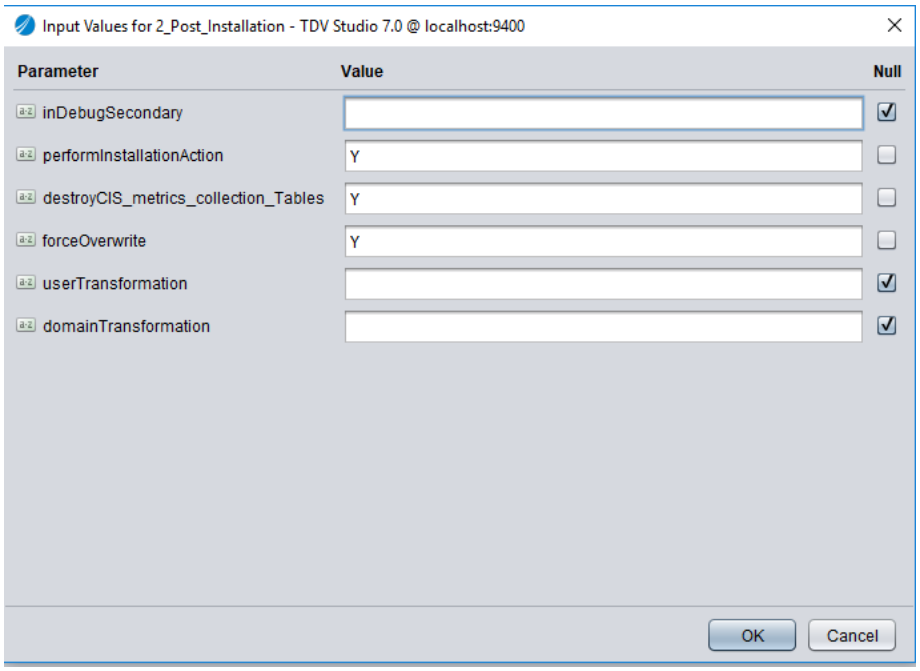
Execute KPImetrics Installation script

1. Execute the Installation to create the database tables

a. ***WARNING: DO NOT RUN THIS IN AN EXISTING ENVRIONMENT. IT WILL DROP ALL KPIMETRICS TABLES***.

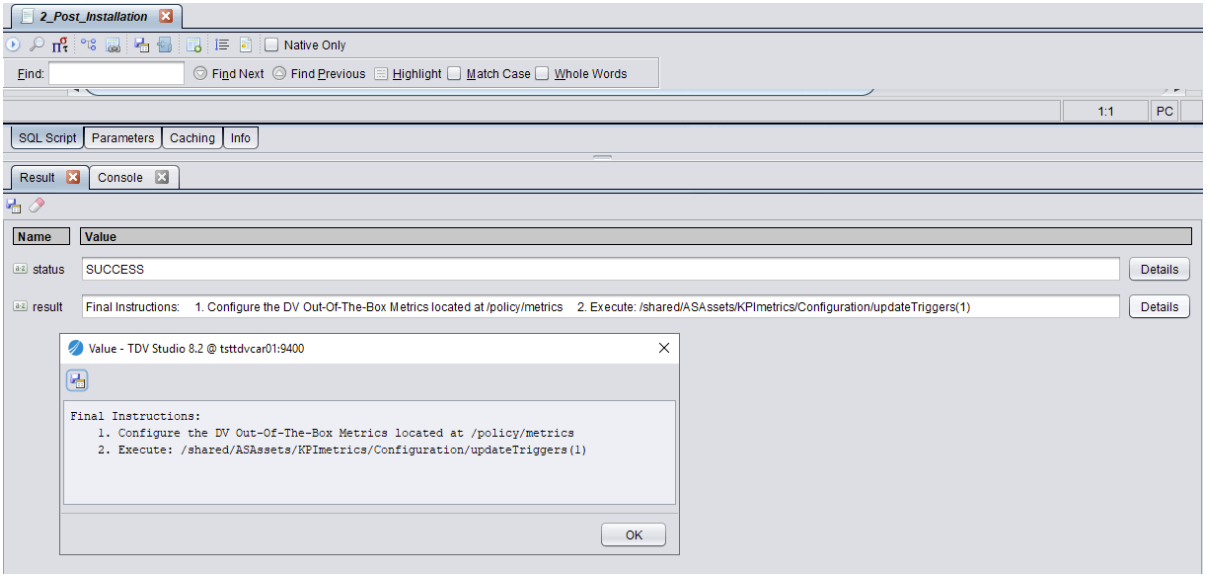
b. /shared/ASAssets/KPImetrics_installation/2_Post_Installation

© Copyright 2000-2017 TIBCO Software Inc.



c. Click OK

d. The status should be “SUCCESS”



© Copyright 2000-2017 TIBCO Software Inc.

Configure TDV Metrics [/policy/metrics]

1. Configure the DV Out-Of-The-Box Metrics located at /policy/metrics
    a. Point to the correctly configured datasource in the Metadata folder:
        i. /shared/ASAssets/KPImetrics/Physical/Metadata
    b. Click “Create or Bind Tables”
        i. Browse and select the the schema of the datasource configured.
        ii. DO NOT “EXECUTE DDL” – the tables already exist.
        iii. Just click OK to accept the configuration.
        iv. Click Save
    c. Request Count Threshold: 1000 is acceptable
    d. How long do you want to keep the metrics data? 1 Day
    e. How often do you want to run the truncate process on expired data? 1 Hour
2. Enable – click enable
    a. Make sure both Buffering Status and Truncate Status is green.

Turn on KPImetrics triggers

1. Execute: /shared/ASAssets/KPImetrics/Configuration/updateTriggers(1)

© Copyright 2000-2017 TIBCO Software Inc.

6 KPImetrics Administration Scenarios

Turn KPI On/Off

This section describes how to turn KPImetrics on and off by simply turning on/off the triggers.

1. Turn OFF KPImetrics triggers
    a. Execute this procedure /shared/ASAssets/KPImetrics/Configuration/updateTriggers
        i. Enable=0
        ii. includeList=null
        iii. excludeList=null
    b. Refresh Studio
    c. Review the triggers status on the Manager tab / Triggers:

Resource	Status	Type	Frequency
kpimetricsTrig_00_CheckMetricsActivityDebug	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_01_Cache_ALL_RESOURCES	Disabled	Timer Event	Every 2 hours at minute
kpimetricsTrig_02_Cache_ALL_USERS	Disabled	Timer Event	Every 2 hours at minute
kpimetricsTrig_03_Cache_LDAP_PERSON	Disabled	Timer Event	Every day at time
kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_06_Cache_LOG_DISK	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_07_Cache_LOG_IO	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_08_Cache_LOG_MEMORY	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST	Disabled	Timer Event	Every 30 minutes
kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS	Disabled	Timer Event	Every day at time
kpimetricsTrig_12_Cache_SYS_CACHES	Disabled	Timer Event	Every 6 hours at minute
kpimetricsTrig_13_Cache_SYS_DATASOURCES	Disabled	Timer Event	Every 12 hours at minute
kpimetricsTrig_14_CheckCISWorkflowStatusFail	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_15_CheckMetricsActivity	Disabled	Timer Event	Every day at time
kpimetricsTrig_16_PurgeHistoryData	Disabled	Timer Event	Every week at day/time
kpimetricsTrig_17_CheckExceedMemoryPercentRequests	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_18_CheckLongRunningRequests	Disabled	System Event	
kpimetricsTrig_19_AllCustom_AccessByUserOverTime	Disabled	Timer Event	Every day at time
kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime	Disabled	Timer Event	Every day at time
kpimetricsTrig_21_AllCustom_ResourceCount_Details	Disabled	Timer Event	Every day at time
kpimetricsTrig_22_AllCustom_ResourceCount_Total	Disabled	Timer Event	Every day at time
kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_24_ValidateUpdateStatusTables	Disabled	Timer Event	Every 30 minutes
kpimetricsTrig_25_ExecuteMetricsSqlControl	Disabled	Timer Event	Every 15 minutes
kpimetricsTrig_30_DBMSScheduler	Disabled	Timer Event	Every 2 hours at minute
kpimetricsTrig_31_DBMSSchedulerError	Disabled	Timer Event	Every 2 hours at minute
kpimetricsTrig_32_DBMSPartitionManager	Disabled	Timer Event	Every day at time
kpimetricsTrig_33_DeleteCollection	Disabled	Timer Event	Every 30 minutes
kpimetricsTrig_34_DBMSRebuildIndexes	Disabled	Timer Event	Every week at day/time
kpimetricsTrig_35_DBMSDeleteCheck	Disabled	Timer Event	Every hour at minute
kpimetricsTrig_40_Cache_METADATA_TABLES	Disabled	Timer Event	Every day at time
kpimetricsTrig_50_MetricsPersistenceFailure	Disabled	System Event	
kpimetricsTrig_51_MetricsTruncateFailure	Disabled	System Event	
kpimetricsTrig_52_MetricsBackupFailure	Disabled	System Event	
kpimetricsTrig_53_MetricsRestoreFailure	Disabled	System Event	

2. Turn ON KPImetrics triggers
    a. Execute this procedure /shared/ASAssets/KPImetrics/Configuration/updateTriggers
        i. Enable=1

© Copyright 2000-2017 TIBCO Software Inc.

ii. includeList=null

iii. excludeList=null

b. Refresh Studio

c. Review the triggers status on the Manager tab / Triggers:

Resource	Status	Type	Frequency	Next Time
kpimetricsTrig_00_CheckMetricsActivityDebug	Disabled	Timer Event	Every hour at minute	
kpimetricsTrig_01_Cache_ALL_RESOURCES	Ready	Timer Event	Every 2 hours at minute	2/25/20 12:00:00 PM
kpimetricsTrig_02_Cache_ALL_USERS	Ready	Timer Event	Every 2 hours at minute	2/25/20 12:00:00 PM
kpimetricsTrig_03_Cache_LDAP_PERSON	Ready	Timer Event	Every day at time	2/26/20 7:00:00 AM
kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES	Ready	Timer Event	Every hour at minute	2/25/20 11:15:00 AM
kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER	Ready	Timer Event	Every hour at minute	2/25/20 11:00:00 AM
kpimetricsTrig_06_Cache_LOG_DISK	Ready	Timer Event	Every hour at minute	2/25/20 11:00:00 AM
kpimetricsTrig_07_Cache_LOG_IO	Ready	Timer Event	Every hour at minute	2/25/20 11:00:00 AM
kpimetricsTrig_08_Cache_LOG_MEMORY	Ready	Timer Event	Every hour at minute	2/25/20 11:00:00 AM
kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST	Disabled	Timer Event	Every 30 minutes	
kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS	Disabled	Timer Event	Every day at time	
kpimetricsTrig_12_Cache_SYS_CACHES	Ready	Timer Event	Every 6 hours at minute	2/25/20 1:00:00 PM
kpimetricsTrig_13_Cache_SYS_DATASOURCES	Ready	Timer Event	Every 12 hours at minute	2/25/20 6:00:00 PM
kpimetricsTrig_14_CheckCISWorkflowStatusFail	Ready	Timer Event	Every hour at minute	2/25/20 11:30:00 AM
kpimetricsTrig_15_CheckMetricsActivity	Ready	Timer Event	Every day at time	2/26/20 12:59:50 AM
kpimetricsTrig_16_PurgeHistoryData	Ready	Timer Event	Every week at day/time	2/29/20 4:30:00 AM
kpimetricsTrig_17_CheckExceedMemoryPercentRequests	Ready	Timer Event	Every hour at minute	2/25/20 11:00:00 AM
kpimetricsTrig_18_CheckLongRunningRequests	Ready	System Event		
kpimetricsTrig_19_AllCustom_AccessByUserOverTime	Ready	Timer Event	Every day at time	2/26/20 12:15:00 AM
kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime	Ready	Timer Event	Every day at time	2/26/20 12:15:00 AM
kpimetricsTrig_21_AllCustom_ResourceCount_Details	Ready	Timer Event	Every day at time	2/26/20 12:15:00 AM
kpimetricsTrig_22_AllCustom_ResourceCount_Total	Ready	Timer Event	Every day at time	2/26/20 12:15:00 AM
kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE	Disabled	Timer Event	Every hour at minute	
kpimetricsTrig_24_ValidateUpdateStatusTables	Ready	Timer Event	Every 30 minutes	2/25/20 11:10:00 AM
kpimetricsTrig_25_ExecuteMetricsSqlControl	Ready	Timer Event	Every 15 minutes	2/25/20 10:50:00 AM
kpimetricsTrig_30_DBMSScheduler	Ready	Timer Event	Every 2 hours at minute	2/25/20 12:30:00 PM
kpimetricsTrig_31_DBMSSchedulerError	Ready	Timer Event	Every 2 hours at minute	2/25/20 11:00:00 AM
kpimetricsTrig_32_DBMSPartitionManager	Ready	Timer Event	Every day at time	2/26/20 12:00:00 AM
kpimetricsTrig_33_DeleteCollection	Disabled	Timer Event	Every 30 minutes	
kpimetricsTrig_34_DBMSRebuildIndexes	Ready	Timer Event	Every week at day/time	3/1/20 3:00:00 AM
kpimetricsTrig_35_DBMSDeleteCheck	Disabled	Timer Event	Every hour at minute	
kpimetricsTrig_40_Cache_METADATA_TABLES	Disabled	Timer Event	Every day at time	
kpimetricsTrig_50_MetricsPersistenceFailure	Ready	System Event		
kpimetricsTrig_51_MetricsTruncateFailure	Ready	System Event		
kpimetricsTrig_52_MetricsBackupFailure	Ready	System Event		
kpimetricsTrig_53_MetricsRestoreFailure	Ready	System Event		

3. Turn ON KPImetrics triggers [TRICK]

a. Usage: In this scenario you have executed the triggers and they are up and running. Now you need to stop all the triggers for some maintenance. When the maintenance is over, you want to restart the triggers but you don't need to or want to wait for triggers 1-3 to perform their operation since the data in their tables is current given that you had a short maintenance window within the same day. The trick is to turn on all of the triggers except 1-3 which you will do manually.

b. Execute this procedure /shared/ASAssets/KPImetrics/Configuration/updateTriggers.

i. The procedure will run very fast since it does not have to load the data for triggers 1-3 as you will put them in the excludeList as shown below.

ii. Enable=1

iii. includeList=null

iv. excludeList=1,2,3

c. In Studio, go to the Manager (tab) → Triggers (screen)

i. Locate the following three triggers and highlight them

© Copyright 2000-2017 TIBCO Software Inc.

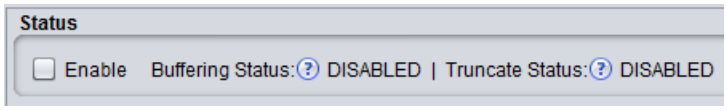
ii. Click on “Change Enabling” button

iii. Now all of the triggers are operational and will execute based on their next time slot.

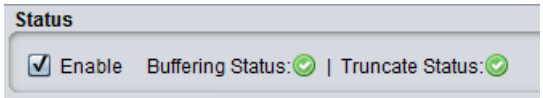
Turn Data Virtualization (DV) metrics On/Off

This section describes how to turn DV metrics on/off.

1. To Stop DV metrics, you must log in as the “admin” user
   a. Open /policy/metrics
   b. Uncheck the “Enable” button
   c. The metrics will indicate they are DISABLED



2. To Start DV metrics, you must log in as the “admin” user
   a. Open /policy/metrics
   b. Check the “Enable” button
   c. Both Buffering Status and Truncate Status should be green.



   d. If they are not, then there is a problem with the metrics configuration. Check the log entries for errors.

Modify Triggers

This section describes how to modify the triggers once they are installed.

1. Open/Edit the resource /shared/ASAssets/KPImetrics/Customize/**defaultTriggersToEnable**
   a. Only modify the ON/OFF settings for each trigger. Leave all other settings alone.
      i. Refer to the following sections for details on each trigger:
         1. Metadata System Triggers and Load Scripts
   b. The current triggers defaulted to OFF are as follows:
      i. **kpimetricsTrig_00_CheckMetricsActivityDebug** – Only turn this on if you suspect that DV metrics is not working properly and you want to debug the DV metrics every hour.
      ii. **kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST** – This trigger is defaulted

© Copyright 2000-2017 TIBCO Software Inc.

to OFF. If you wish to perform SQL parsing on the request description SQL statement to parse out the table and column resources used in the SQL then turn this trigger on. There is quite a bit of overhead associated with this trigger.

iii. **kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS** – Only turn this on if you get a code update from the Open Source site and there were changes to the SQL Parser code.

iv. **kpimetricsTrig_17_CheckExceedMemoryPercentRequests** – You may choose to keep this off in lower-level environments but turn it on in PROD environments.

v. **kpimetricsTrig_18_CheckLongRunningRequests** – You may choose to keep this off in lower-level environments but turn it on in PROD environments.

vi. **kpimetricsTrig_33_DeleteCollection** – This trigger is defaulted to OFF and is deprecated as of 2020Q100 release. This trigger deletes unwanted rows in the metrics_resources_usage based on applicable rows found in METRICS_JOB_FILTERS. It also deletes rows in metrics_requests where a corresponding row does not exist in metrics_resources_usage. As part of the normal KPImetrics operation, the trigger kpimetricsTrig_30_DBMSScheduler runs a procedure that will transfer metrics collection data and delete rows from the metrics_resources_usage and metrics_requests collection tables. If this trigger/procedure is operating without issues and keeping up with deletes, then it is possible to turn off the trigger: **kpimetricsTrig_33_DeleteCollection**.

vii. **kpimetricsTrig_35_DBMSDeleteCheck** – This trigger is used to monitor when TDV metrics executes the delete statements against the collection tables. It is optional but may provide value in terms of getting an idea what is being executed by TDV metrics.

viii. **kpimetricsTrig_40_Cache_METADATA_TABLES** – This trigger is defaulted to OFF. This trigger executes the KPI Metadata strategy. It is intensive which is why it only runs once a day. It could be set to run less often if metadata does not change that often. It should never run more than once a day as it will put load on the DV server. Refer to “KPImetadata Configuration Guide v1.3.pdf” for more information about configuration and usage. Before turning this on, the following procedure must be configured: /shared/ASAssets/KPImetrics/Customize/pqInsert_METADATA_Constants.

2. Execute this procedure /shared/ASAssets/KPImetrics/Configuration/updateTriggers
   a. Enable=1
   b. includeList=null
   c. excludeList=null

© Copyright 2000-2017 TIBCO Software Inc.

3. Refresh Studio
4. Review the triggers status on the Manager tab / Triggers to ensure your trigger changes were enforced.

Perform Oracle Database Maintenance on Collection Tables

This section outlines how to perform maintenance on the Oracle collection tables in order to regain the tablespace.

1. Stop Data Virtualization metrics
2. Execute the /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/**P_METRICS_ALL_TABLES_exec** to process the existing rows in the collection tables
3. Execute the following commands directly against Oracle

   ALTER TABLE metrics_sessions ENABLE ROW MOVEMENT;

   ALTER TABLE metrics_sessions SHRINK SPACE CASCADE;

   ALTER TABLE metrics_sessions DEALLOCATE UNUSED KEEP 50;

   ALTER TABLE metrics_requests ENABLE ROW MOVEMENT;

   ALTER TABLE metrics_requests SHRINK SPACE CASCADE;

   ALTER TABLE metrics_requests DEALLOCATE UNUSED KEEP 50;

   ALTER TABLE metrics_resources_usage ENABLE ROW MOVEMENT;

   ALTER TABLE metrics_resources_usage SHRINK SPACE CASCADE;

   ALTER TABLE metrics_resources_usage DEALLOCATE UNUSED KEEP 50;

Perform SQL Server Database Maintenance on Collection Tables

This section outlines how to perform maintenance on the SQL Server collection tables in order to regain the tablespace.

1. Using SQL Server Management Console, run the following script to view the tablespace sizes

```
SELECT name AS NameOfFile, size/128.0 as TotalSizeInMB,
        CAST(FILEPROPERTY(name, 'SpaceUsed' )AS int)/128.0 AS SpacesUsedInMB,
        size/128.0 -CAST(FILEPROPERTY(name, 'SpaceUsed' )AS int)/128.0 AS AvailableSpaceInMB
FROM dbo.SYSFILES
GO
```

2. Example results:
   a. Filegroup results:

© Copyright 2000-2017 TIBCO Software Inc.

Results | Messages

	NameOfFile	TotalSizeInMB	SpacesUsedInMB	AvailableSpaceInMB
1	CIS_KPI1	22.250000	8.812500	13.437500
2	CIS_KPI_log	4180.187500	4151.312500	28.875000
3	METRICS_DATA_COLL	2743.062500	1859.937500	883.125000
4	METRICS_DATA_HIST	2019.812500	2019.812500	0.000000
5	METRICS_DATA_IDX	2920.875000	2920.875000	0.000000

b. File system results:

Name	Date modified	Type	Size
CIS_KPI_log_1.ldf	1/21/2020 11:09 AM	SQL Server Databa...	4,280,512 KB
CIS_KPI_primary_1.ndf	1/20/2020 10:45 AM	SQL Server Databa...	22,784 KB
METRICS_DATA_COLL_1.ndf	1/20/2020 2:47 PM	SQL Server Databa...	2,808,896 KB
METRICS_DATA_HIST_1.ndf	1/21/2020 10:46 AM	SQL Server Databa...	2,068,288 KB
METRICS_DATA_IDX_1.ndf	1/21/2020 10:46 AM	SQL Server Databa...	2,990,976 KB

3. Typically, the SQL Server log is what is growing the most and must be managed. The following SQL can be run from SQL Server Management Console to shrink the log and the log file thus freeing up disk space.
   a. Note that the log file group will be different in your environment:

```
DBCC SHRINKFILE (N'DVKPI_log', 0, TRUNCATEONLY)
GO
```

4. Example results after execute “DBCC shrinkfile”:
   a. Filegroup results:

Results | Messages

	NameOfFile	TotalSizeInMB	SpacesUsedInMB	AvailableSpaceInMB
1	CIS_KPI1	22.250000	8.812500	13.437500
2	CIS_KPI_log	1.000000	0.367187	0.632813
3	METRICS_DATA_COLL	2743.062500	1859.937500	883.125000
4	METRICS_DATA_HIST	2019.812500	2019.812500	0.000000
5	METRICS_DATA_IDX	2920.875000	2920.875000	0.000000

   b. File system results:
      i. In this example, the log file shrunk from 4GB down to 1 MB after executing the DBCC shrinkfile command. This was after DV metrics were turned off and the DV server was recycled so that the connection to SQL Server was released thus allowing SQL Server to release the logs.

© Copyright 2000-2017 TIBCO Software Inc.

Name	Date modified	Type	Size
CIS_KPI_log_1.ldf	1/21/2020 11:19 AM	SQL Server Databa...	1,024 KB
CIS_KPI_primary_1.ndf	1/20/2020 10:45 AM	SQL Server Databa...	22,784 KB
METRICS_DATA_COLL_1.ndf	1/20/2020 2:47 PM	SQL Server Databa...	2,808,896 KB
METRICS_DATA_HIST_1.ndf	1/21/2020 10:46 AM	SQL Server Databa...	2,068,288 KB
METRICS_DATA_IDX_1.ndf	1/21/2020 10:46 AM	SQL Server Databa...	2,990,976 KB

Configure Third Party Tool Access

This section outlines how to configure third party tool access for reporting tools such as Cognos, Spotfire, MicroStrategy or others.

1. Download the Data Virtualization (DV) ODBC 7 Drivers and install on the client host machine
2. Configure an ODBC data source
    a. DSN Name: Provide different connections to different DV instances
        i. DV_KPIMETRICS_DEV
        ii. DV_KPIMETRICS_UAT
        iii. DV_KPIMETRICS_PROD
    b. Host – hostname of the DV target instance
    c. Port – port number of the DV target instance (e.g. 9401)
    d. User Name – the user name or service account to use
    e. Password – the password for the user name or service account
    f. Domain – the domain name such as “composite” or “ldap”
    g. Data source – the data source will be “ASAssets”
    h. Catalog – can leave this blank
    i. Local/Code Page – can leave this blank

Get the Current Row Distribution for the History Tables/Partitions

This section describes how to get the row distribution for the three history tables and their partitions.

Generically:
/services/databases/ASAssets/KPImetrics/metrics/metrics_history_tables_row_distribution

© Copyright 2000-2017 TIBCO Software Inc.

FAQ

This section provides an FAQ.

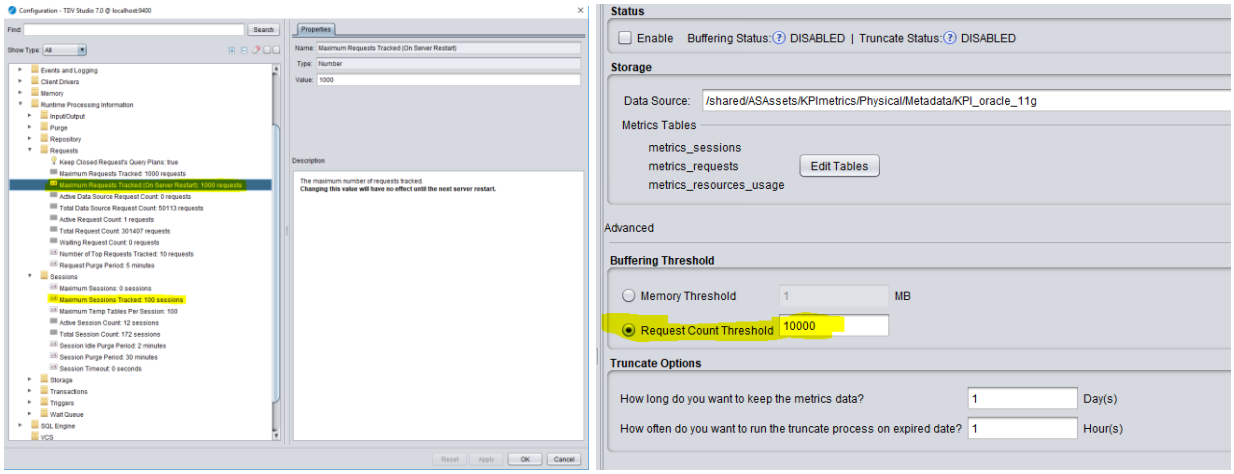
Q1: Does TDV metrics use memory or temp files to store the metrics before writing it out to the database?

A1: It uses a memory buffer. The amount of memory varies based on the Policy Metrics settings.

Q2: When using SQL Server for the database, is it advantageous to configure BCP for better performance inserting into collection tables?

A2: All information is kept in memory and a JDBC prepared statement is used to insert into the database in batches.

Q3: Is there a correlation between the administration configuration parameter "Maximum Requests Tracked" and the policy metrics setting "Request Count Threshold"?



Admin Config: Maximum Requests Tracked – 1000 requests

Admin Config: Maximum Sessions Tracked – 100 sessions

Policy Metrics: Request Count Threshold - 10000

A3: No there is not a correlation between them. Metrics will keep the information in memory until the threshold is reached and write them to the database.

© Copyright 2000-2017 TIBCO Software Inc.

Q4: Will I lose transactions and sessions because metrics is set to 10,000 and Max Request Tracked is set to 1000? Any correlation?

A4: No. There will not be any loss of information. All requests are kept in memory until the threshold is reached. The only correlation between them if smaller Max Request Tracked, then it will take longer to fill the metrics buffer to reach Metrics request threshold.

Q5: What can be expected in terms of memory usage if Policy metrics Request Count Threshold is set to 10000?

A5: It's really based on how big the size of session, request and resource events are. The real memory usage may differ. One option is to use memory threshold.

© Copyright 2000-2017 TIBCO Software Inc.

7 KPImetrics Resources

Configuration Resources

This section outlines the resources that are used for configuration of KPImetrics.

KPI Version Overview

Location: /shared/ASAssets/KPImetrics

This section lists the version and “How to Install” procedures.

Configuration Name	Description
getKPIVersion	Returns the version of KPI in the format: YYYY.Q[1-4] e.g. 2017.4 If a patch is release within the Quarter then it would be 2017.401
README	Provides a brief description of the published views.
RELEASE_NOTES	Provides a description of what changes per release.

Configuration Customize Folder Overview

Location: /shared/ASAssets/KPImetrics/Customize

This section lists all of the configuration customization resources that the user will configure.

Configuration Name	Description
commonValues	Script to store default purge time values and data source information.
defaultTriggersToEnable	This view contains a series of UNION statements for each trigger that exists in the KPImetrics. It provides information on the following: triggerName, triggerNumber, recommendation, executeImmediate, isCache and cacheTableName. Example row: 'kpimetricsTrig_01_Cache_ALL_RESOURCES' triggerName, 1 triggerNumber , 'ON' recommendation, 1 executeImmediate, 0 isCache, '' cacheTableName
LDAP_PERSON	This is the view that integrates and customizes the LDAP1 and LDAP2 datasources.
pqInsert_METRICS_EVENT_REGISTRATION	This procedure is used for either initial load or maintenance. If a row already exists, it does not update it. It simply bypasses it. This means that you can run this procedure as many times as you want and not impact existing rows. It does not delete or unsubscribe requester events. To delete a subscription, invoke pMetricsEventRegistrationUnsubscribe(). A subscription in the METRICS_EVENT_REGISTRATION table consists of a unique record for the combination of SUBSCRIBER_EMAIL, GROUP_NAME, ENVIRONMENT_TYPE, EVENT_TYPE and REQUESTER_EMAIL. This is the metrics event subscription table: METRICS_EVENT_REGISTRATION

© Copyright 2000-2017 TIBCO Software Inc.

	SUBSCRIBER_EMAIL PK userid email or a group email alias. Who the email alert will be sent to. GROUP_NAME PK Group name subscribing to. When an alert occurs for a user the groups will be checked and cross-referenced with this registered group. The group [all] is a composite group and a catch-all for any user belonging to this composite group. ENVIRONMENT_TYPE PK Register for all environments [ALL] or a certain environment type [DEV1, CIT1, SIT1, UAT, TT, PROD] EVENT_TYPE PK [LONG_RUNNING| EXCEEDED_MEMORY| INACTIVITY| WORKFLOW_FAILURE| DBMS_SCHEDULER_ERROR| PURGE_HISTORY] REQUESTER_EMAIL PK Primary requester email. REQUESTER_FIRST_NAME Primary requester first name. REQUESTER_LAST_NAME Primary requester last name. EXCLUDE_TEXT Pipe separated list of text phrases that would appear in the SYS_REQUESTS SQL description field and signify exclusion of this event if the text is found in the SQL description.
pqInsert_METRICS_JOB_tables	This procedure is used for either initial load or maintenance. It deletes all of the rows and then inserts them new. For maintenance, simply make a change to the script and re-execute in each environment. Perform the load/maintenance on the following tables: METRICS_JOB_ENVIRONMENTS, METRICS_JOB_FILTERS
pqInsert_METADATA_Constants	This procedure gets invoked by /Physical/Metadata/System/ClusterSafeCache/Cache_METADATA_TABLES on a regular schedule so that this data stays in sync with the other metadata.

Configuration Folder Overview

Location: /shared/ASAssets/KPImetrics/Configuration

This section lists all of the configuration procedures that have been defined for the KPImetrics module. These scripts provide functionality for setup/configuration of the KPImetrics module.

Configuration Name	Description
getQueryResponseTime	This procedure executes the passed in query and calculates the average time taken to retrieve the first row of data.
pMaintenanceUpdateHostnamePort	This procedure is used to convert the values for NODE_HOST/nodehost and NODE_PORT/nodeport from one value to another. It is “highly” unlikely that this procedure would ever be used but it is provided nonetheless for unforeseen circumstances. For each view found in /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction perform the updated operation if they contain both NODE_HOST/nodehost and NODE_PORT/nodeport. This is a dynamic procedure so it will pick up whatever is in that path.

© Copyright 2000-2017 TIBCO Software Inc.

	Note: If you (1) migrate from one host to another host or (2) migrate from one DV instance to another DV instance on the same host, you should not modify the data. The nodehost and nodeport along with requestid are needed to insure a unique row. When moving to a new DV instance, the requestid may be repeated.
pMetricsEventRegistrationList	This procedure is used to "LIST" to a metrics event registration. A subscription in the METRICS_EVENT_REGISTRATION table that consists of a unique record for the combination of SUBSCRIBER_EMAIL, GROUP_NAME, ENVIRONMENT_TYPE, EVENT_TYPE and REQUESTER_EMAIL.
pMetricsEventRegistrationSubscribe	This procedure is used to "SUBSCRIBE" to a metrics event registration. A subscription in the METRICS_EVENT_REGISTRATION table consists of a unique record for the combination of SUBSCRIBER_EMAIL, GROUP_NAME, ENVIRONMENT_TYPE, EVENT_TYPE and REQUESTER_EMAIL. The column EVENT_TYPE is one of: [LONG_RUNNING| EXCEEDED_MEMORY| INACTIVITY| WORKFLOW_FAILURE| DBMS_SCHEDULER_ERROR]
pMetricsEventRegistrationUnsubscribe	This procedure is used to "UNSUBSCRIBE" to a metrics event registration. A subscription in the METRICS_EVENT_REGISTRATION table that consists of a unique record for the combination of SUBSCRIBER_EMAIL, GROUP_NAME, ENVIRONMENT_TYPE, EVENT_TYPE and REQUESTER_EMAIL.
rebindPhysicalAbstraction	The 1st step of this procedure is used to rebind all of the resources (Views) in /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction to the KPI_<database_type> folder as configured in commonValues. All views above the /Abstraction layer will be redirected to use the correct data source. This is a one-time configuration done during setup. The 2nd step is used to optimize various views and procedures as defined by commonValues.viewOptimizationPathList. Some examples of optimization: SQL SERVER: ex 1. {OPTION IGNORE_TRAILING_SPACES="TRUE", CASE_SENSITIVE="FALSE"} ex 2. CAST(DATEDIFF('DAY', requestdate, CURRENT_DATE)/31.00 AS DECIMAL(19,2)) requestdatemonths, CAST(DATEDIFF('DAY', requestdate, CURRENT_DATE) AS DECIMAL(19,0)) requestdatedays, ex 3. FROM /shared/ASAssets/KPImetrics/AllCustomReports/ ResourceCount_DetailsRT_sqlserver ex 4. FROM /shared/ASAssets/KPImetrics/AllCustomReports/ ResourceCount_TotalRT_sqlserver ORACLE: ex 1. {OPTION IGNORE_TRAILING_SPACES="FALSE", CASE_SENSITIVE="TRUE"} ex 2. CAST(ROUND(MONTHS_BETWEEN(CURRENT_DATE, requestdate),2) AS DECIMAL(19,2)) requestdatemonths, CAST(DAYS_BETWEEN(CURRENT_DATE, requestdate) AS DECIMAL(19,0)) requestdatedays, ex 3. FROM /shared/ASAssets/KPImetrics/AllCustomReports/ ResourceCount_DetailsRT_oracle

© Copyright 2000-2017 TIBCO Software Inc.

	ex 4. FROM /shared/ASAssets/KPImetrics/AllCustomReports/ ResourceCount_TotalRT_oracle
rebindPhysicalDatabaseType	Rebind the folder /shared/ASAssets/KPImetrics/Physical/Physical/KPI_<database_type> [KPI_oracle, KPI_sqlserver]. Rebind from whatever the current folder is currently pointing to and modify to the new catalog/schema path for tables and procedures. This is required when the catalog and/or the schema name are changed from the original setting. This may be required on the initial setup in the development environment. This may be required if upper environments have a different catalog or schema from the DEV environment although this is not recommended. This procedure is used when the schema and catalog for Oracle is not 'DVKPI' or SQLServer is not 'DVKPI/dbo'. Configure the following /shared/ASAssets/KPImetrics/Customize/commonValues.dataSourceName PRIOR to running this procedure.
updateImpactedResources	Provides a way to iterate through /shared/ASAssets/KPImetrics and attempt to fix any impacted resources due to an anomaly in the DV repository.
updateTriggers	This procedure is used to enable and disable the triggers based on the stored in /Customize/defaultTriggersToEnable(). When enabling the triggers, it will place a record in METRICS_SQL_CONTROL for Cach_ALL_RESOURCES and Cache_ALL_USERS for each node of the cluster which inidicates that those 2 procedures need to be executed prior to the dependents requiring the data. The following processes will look at METRICS_SQL_CONTROL to determine if the procedures need executing: pExecuteMetricsSqlControl – will execute both and is triggered by kpimetricsTrig_25_ExecuteMetricsSqlControl. Cache_ALL_RESOURCES gets executed by dependents: Cache_ALL_RESOURCES Cache_METRICS_SQL_REQUEST_EXEC Cache_METRICS_SQL_REQUEST_EXEC_ADHOC Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS Cache_METRICS_SQL_RESOURCE_LINEAGE pMETRICS_ALL_TABLES_exec Cache_ALL_USERS gets executed by dependents: Cache_ALL_USERS pMETRICS_ALL_TABLES_exec

© Copyright 2000-2017 TIBCO Software Inc.

Published Resources

This section outlines the resources that are published under the ASAssets virtual database to expose metrics data. Resources are organized under catalogs and schemas based upon their functionality.

Please review the document "***KPImetrics Data Dictionary vX.Y.pdf***" for details about published tables, procedures and columns.

KPImetricsAdmin Definitions

This section outlines the resources that are published under the ASAssets virtual database to expose administration functions.

configurations.updateTriggers Procedure

Provides the ability to turn on/off triggers programmatically.

Column Name	Column Type	Definition
enable	SMALLINT	IN
includeList	VARCHAR(255)	IN
excludeList	VARCHAR(255)	IN
operation	VARCHAR(255)	OUT
prevStatus	VARCHAR(10)	OUT
currStatus	VARCHAR(10)	OUT
triggerNumber	INTEGER	OUT
triggerAction	VARCHAR(3)	OUT
excuteImmediate	INTEGER	OUT
rowsInserted	BIGINT	OUT
duration	VARCHAR(25)	OUT
triggerName	VARCHAR(255)	OUT
triggerPath	VARCHAR(1024)	OUT
message	VARCHAR(1200)	The exception message if a request was not successful.

Data Sources

This section outlines the data sources created, populated and used by KPImetrics project.

Metadata Data Source for LDAP

Location:

/shared/ASAssets/KPImetrics/Physical/Metadata/LDAP1

/shared/ASAssets/KPImetrics/Physical/Metadata/LDAP2

The data source LDAP1 and LDAP2 are LDAP data sources that connects to a client's corporate LDAP directory to lookup user information and their relation to client hierarchy. The data source's URL property should be modified to allow the data source to successfully connect to and query the target LDAP directory server. The LDAP structure may vary and could use LDAP person, organizationalPerson, inetOrgPerson or some other customer LDAP

© Copyright 2000-2017 TIBCO Software Inc.

table. Determine the correct table to find person data and introspected it under this data source for LDAP integration to function successfully.

There are two data sources provided out of the box in the event there are two DV domains configured. Add additional data sources if there are more than two domains. Each data source directly corresponds to a DV domain.

It is strongly recommended that this data source should not use the same LDAP account as the DV server uses to authenticate LDAP users. This may result in the LDAP account being locked if the data source's credentials are not updated when the account's password is changed.

Metadata Data Source for CPUAndMemChecker

Location: /shared/ASAssets/KPImetrics/Physical/Metadata/CPUAndMemChecker

CPUAndMemChecker custom java procedure is used to capture system level CPU and Memory usage at the operating system level. On a linux server installation, CPUAndMemChecker invokes two shell scripts (KPImetricsTopCommandGrepCpu_linux7.sh and KPImetricsFreeMemCommand_linux7.sh) to execute 'top' and 'free' commands to returns CPU percentage, used memory and available memory. Windows installations use a couple of powershell scripts (KPImetricsCpuUtilization.ps1 and KPImetricsMemUtilization.ps1) to perform the same capability.

The CPUAndMemChecker procedure is invoked by Cache_CIS_SYSTEM_RESOURCES script and inserts the results in METRICS_CIS_SYSTEM_RESOURCES table of the KPImetrics data source.

The CPUAndMemChecker procedure exposes one procedure that has following parameters:

Parameter Name	Direction	Description
debug	IN	Y=debug values written to cs_server.log. N=no debug.
cpuScriptNameOrCommand	IN	Path to script to return CPU utilization. Windows example: powershell.exe -file C:\CIS7\bin\KPImetricsCpuUtilization.ps1 UNIX example: /CIS7/bin/KPImetricsTopCommandGrepCpu_linux7.sh
memScriptNameOrCommand	IN	Path to script to return memory utilization. Windows example: powershell.exe -file C:\CIS7\bin\KPImetricsMemUtilization.ps1 UNIX example: /CIS7/bin/KPImetricsFreeMemCommand_linux7.sh
cpuUsedPercent	OUT	Average CPU utilization percentage reported by the server's operating system

© Copyright 2000-2017 TIBCO Software Inc.

memoryUsedMb	OUT	Used memory in Megabytes reported by the server's operating system
memoryAvailMb	OUT	Available memory in Megabytes reported by the server's operating system

Metadata Data Source Tables and Procedures

Location: /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_<database_type>_<version>

The KPImetrics module provides data source for all currently supported storage database platforms under /shared/ASAssets/KPImetrics/Physical/Metadata.

Currently the KPImetrics module includes the following KPImetrics data sources

- /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_<version>
- /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_<version>

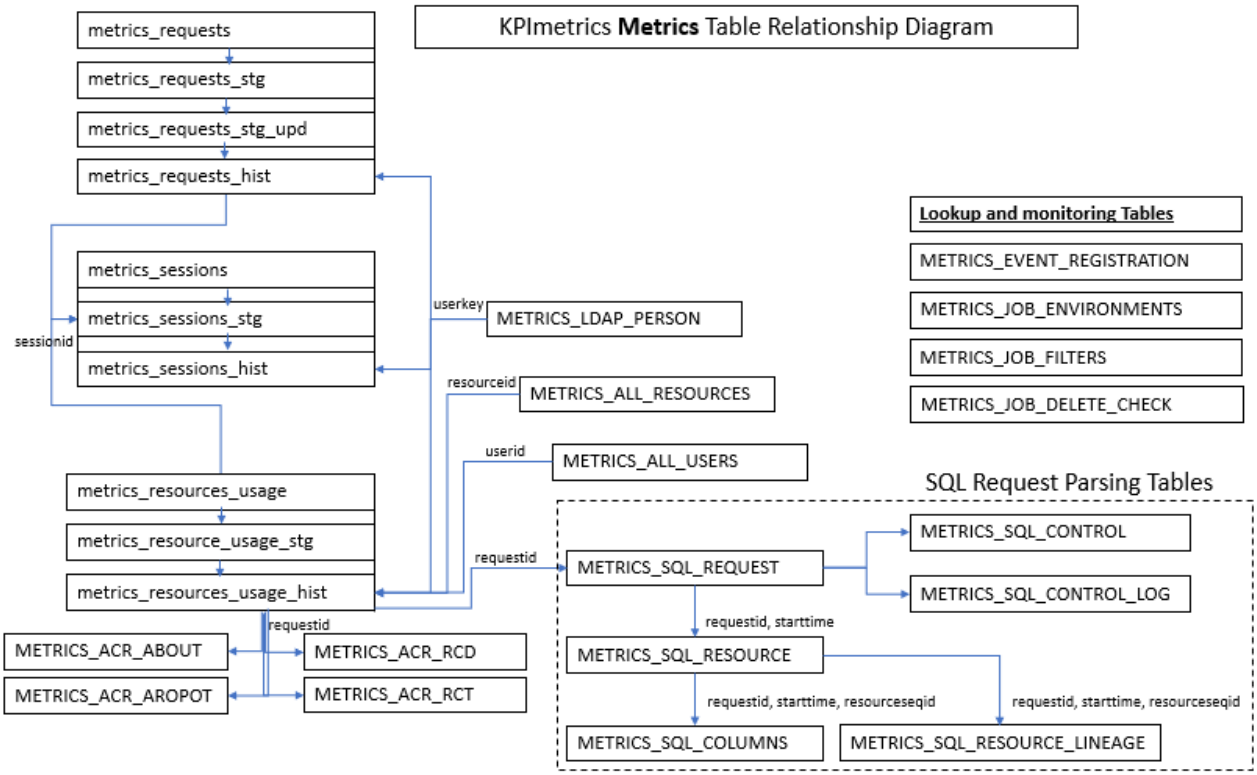
During deployment, the KPImetrics module must be configured to use the data source appropriate for the target KPImetrics database platform. The following instructions refer to this data source at the KPImetrics data source for simplicity

The KPImetrics data source is used to capture

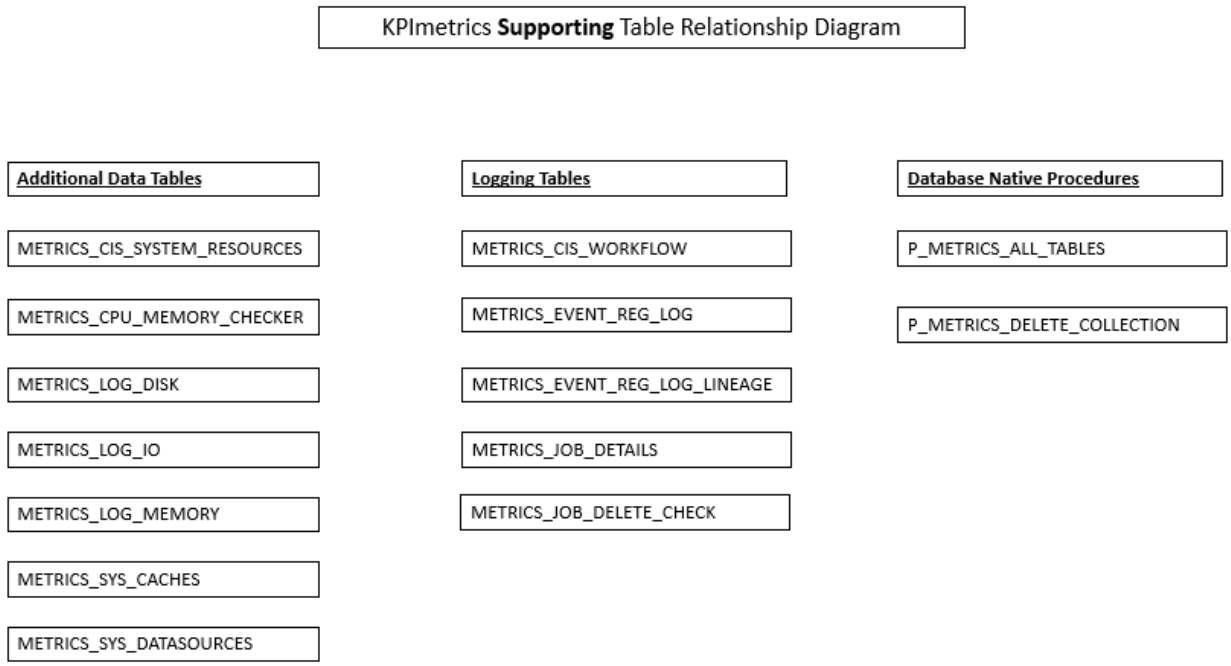
- Historical server metrics captured using incremental caching
- Some pre-processed data for final reporting

© Copyright 2000-2017 TIBCO Software Inc.

KPImetrics Metrics Table Relationship Diagram

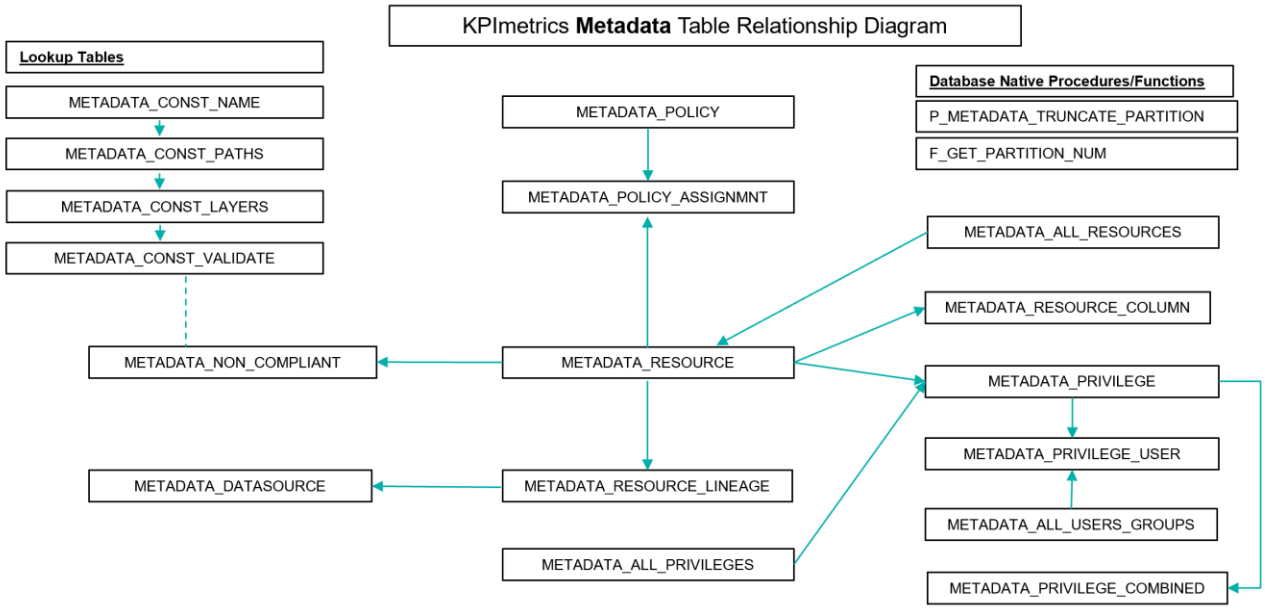


KPImetrics Supporting Table Relationship Diagram



© Copyright 2000-2017 TIBCO Software Inc.

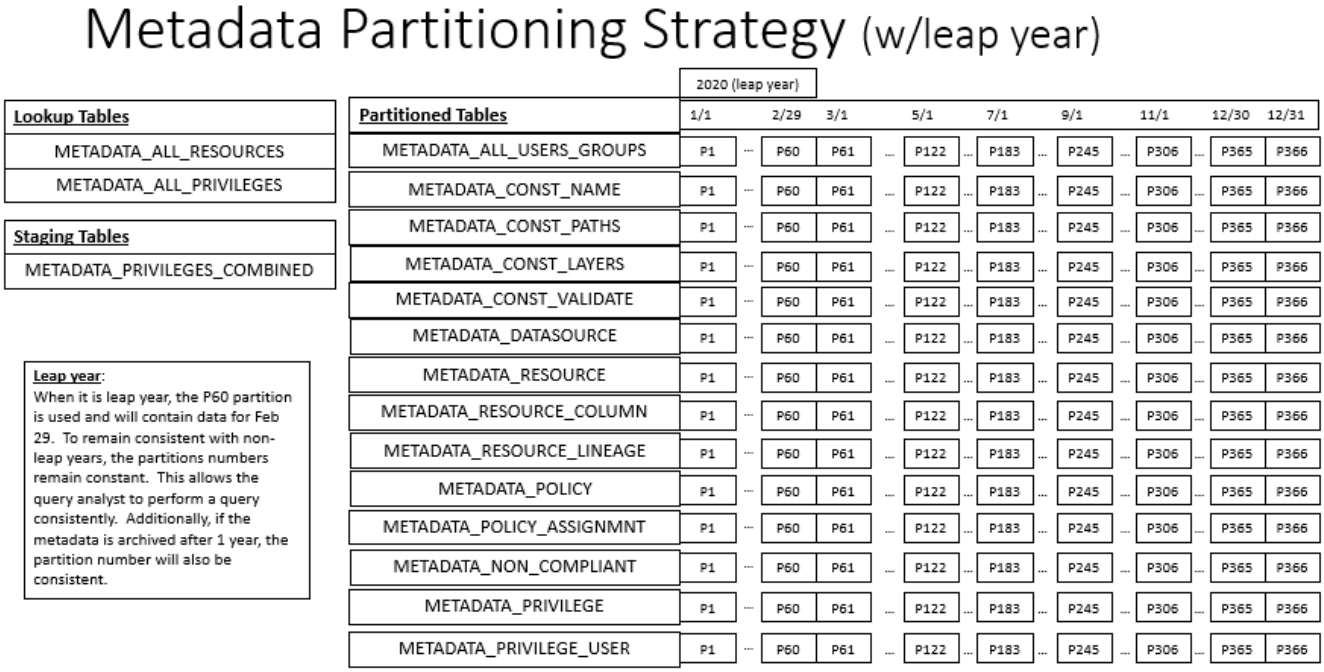
KPImetrics Metadata Table Relationship Diagram



KPI Metrics Metadta Partitioning Strategy (w/leap year)

Leap year:

When it is leap year, the P60 partition is used and will contain data for Feb 29. To remain consistent with non-leap years, the partitions numbers remain constant. This allows the query analyst to perform a query consistently. Additionally, if the metadata is archived after 1 year, the partition number will also be consistent.



Metadata Partitioning Strategy (w/leap year)

Lookup Tables
METADATA_ALL_RESOURCES
METADATA_ALL_PRIVILEGES

Staging Tables
METADATA_PRIVILEGES_COMBINED

Leap year: When it is leap year, the P60 partition is used and will contain data for Feb 29. To remain consistent with non-leap years, the partitions numbers remain constant. This allows the query analyst to perform a query consistently. Additionally, if the metadata is archived after 1 year, the partition number will also be consistent.

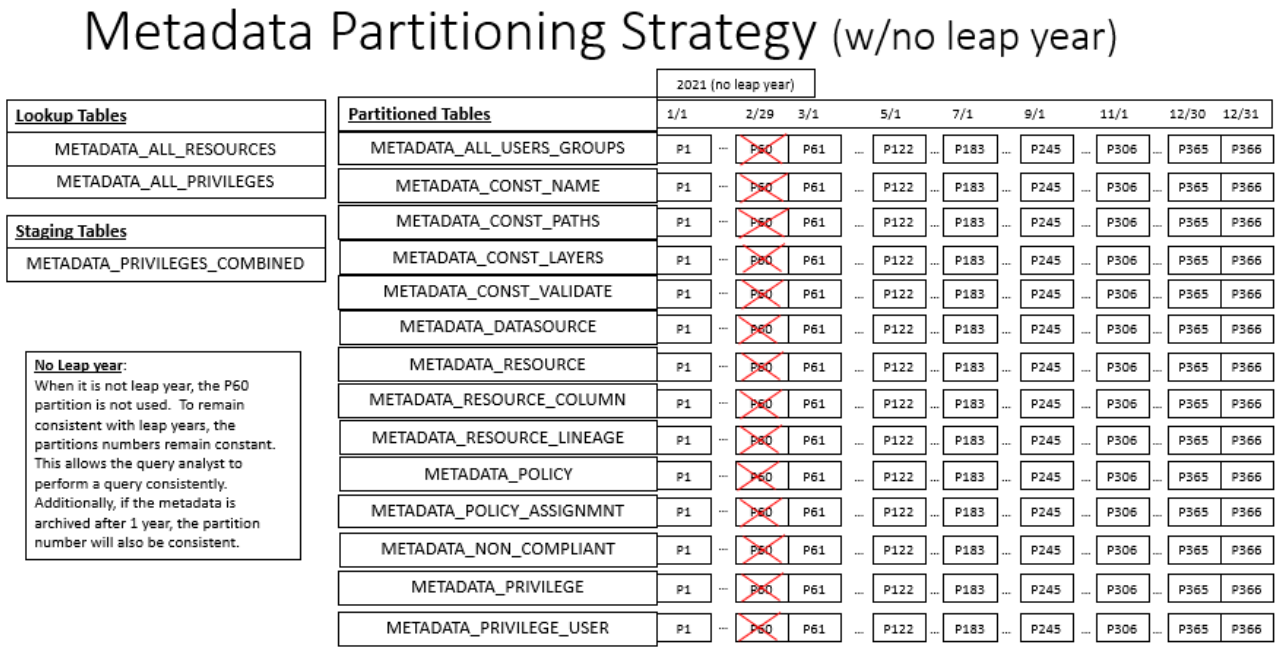
	2020 (leap year)													
Partitioned Tables	1/1		2/29	3/1		5/1		7/1		9/1		11/1		12/30	12/31
METADATA_ALL_USERS_GROUPS	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_CONST_NAME	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_CONST_PATHS	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_CONST_LAYERS	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_CONST_VALIDATE	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_DATASOURCE	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_RESOURCE	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_RESOURCE_COLUMN	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_RESOURCE_LINEAGE	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_POLICY	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_POLICY_ASSIGNMNT	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_NON_COMPLIANT	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_PRIVILEGE	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366
METADATA_PRIVILEGE_USER	P1	...	P60	P61	...	P122	...	P183	...	P245	...	P306	...	P365	P366

© Copyright 2000-2017 TIBCO Software Inc.

KPI Metrics Metadta Partitioning Strategy (w/no leap year)

No Leap year:

When it is not leap year, the P60 partition is not used. To remain consistent with leap years, the partitions numbers remain constant. This allows the query analyst to perform a query consistently. Additionally, if the metadata is archived after 1 year, the partition number will also be consistent.



The following tables have been created in DVKPI schema to capture the required data.

Table Name	Description
P_METRICS_ALL_TABLES	This procedure is used by DV to execute the native database data transfer procedure which formats and moves data from the metrics collection tables to the metrics history tables on a scheduled basis. TRIGGER: /KPImetrics/Physical/Physical/KPI_oracle/kpimetricsTrig_30_DBMSScheduler_KPI → /KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec → /Physical/Physical/Abstraction/P_METRICS_ALL_TABLES → **Oracle Lineage**: PROCEDURES: /KPImetrics/Physical/Physical/KPI_oracle/P_METRICS_ALL_TABLES → /KPImetrics/Physical/Metadata/KPI_oracle_11g/<schema>/P_METRICS_ALL_TABLES **SQL Server Lineage**: PROCEDURES: /KPImetrics/Physical/Physical/KPI_sqlserver/P_METRICS_ALL_TABLES → /KPImetrics/Physical/Metadata/KPI_sqlserver_2016/<catalog>/<schema>/ P_METRICS_ALL_TABLES

© Copyright 2000-2017 TIBCO Software Inc.

P_METRICS_DELETE_COLLECTION	This procedure I sused by DV to execute native database SQL for deleting the collection table data. TRIGGER: /Physical/Physical/KPI_oracle/kpimetricsTrig_33_DeleteCollection → /Physical/Metadata/System/ClusterSafeCache/pMETRICS_DELETE_COLLECTION_exec → /Physical/Physical/Abstraction/P_METRICS_DELETE_COLLECTION → **Oracle Lineage**: PROCEDURES: /Physical/Physical/KPI_oracle/P_METRICS_DELETE_COLLECTION → /Physical/Metadata/KPI_oracle_11g/<schema>/P_METRICS_DELETE_COLLECTION **SQL Server Lineage**: PROCEDURES: /Physical/Physical/KPI_sqlserver/P_METRICS_DELETE_COLLECTION → /Physical/Metadata/KPI_sqlserver_2016/<catalog>/<schema>/ P_METRICS_DELETE_COLLECTION
P_METADATA_TRUNCATE_PARTITION	This database procedure is used to truncate the partitions based on a start date and an end date. It is by cache_METADATA_TABLES via this interface /KPImetrics/Physical/Physical/Abstraction/P_METADATA_TRUNCATE_PARTITION. startDate = loadDate; endDate – A calculation based on loadDate and commonValues.purgeMetadata as described below: There are 366 partitions allowing 1 extra for leap year. Calcuate the total partitions - number of days to keep history for. The startDate and endDate act as a rolling window into the 366 partitions. The value for purgeMetadata comes from commonValues. purgeMetadata. SELECT DATEADD('day', 366-purgeMetadata, loadDate) INTO endDate from DUAL; CALL ../Physical/Physical/Abstraction/P_METADATA_TRUNCATE_PARTITION(debug, dataSourceSchemaName, tableName, startDate, endDate, outValue, partitionList, debugOutput);
F_GET_PARTITION_NUM	This is a database procedure that returns the dat of the year partition number based on the timestamp input. There are 366 days in the year which accounts for leap year. Feb 29 of a leap year is always partition number 60. Mar 1 is always 61. Dec 31 is always 366. In non-leap years, partition 60 will be empty. The partition number remain costant across leap and non-leap years so that queries are preserved accurately. This procedure is accessible from as a custom function via the following procedure /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction/F_GET_PARTITION_NUM This "function" may be used in queries to increase performance by eliminating partitions. *How to query a metadata view and utilize the power of partitioning using partition elimination*? The answer is to include the partition column in each query as well as case sensitivity settings. The partition number represents a day of the year and can be dynamically determined using the F_GET_PARTITION_NUM function. The input must be a timestamp. A string may be CAST as a TIMESTAMP. select * from /shared/ASAssets/KPImetrics/Application/metadata/vMetadataResource where partition >= F_GET_PARTITION_NUM(CAST('2020-04-17 00:00:00' AS TIMESTAMP)) and partition <= F_GET_PARTITION_NUM(CAST('2020-04-19 00:00:00' AS TIMESTAMP)) The range of partitions includes: [108,109,110]. All other partitions will be eliminated from the query thus improving performance.

© Copyright 2000-2017 TIBCO Software Inc.

METRICS_ACR_ABUOT	The cache table acronym "METRICS_ACR_ABUOT" must be short and thus it stands for "All Custom Reports Access By User Over Time". It is cached once a day from the real-time query "ACR_AccessByUserOvertime". It is access from AllCustomReports.AccessByUserOvertime. TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_19_AllCustom_AccessByUserOverTime → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_AllCustom_AccessByUserOverTime → READ: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ACR_AccessByUserOverTime → INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_ACR_ABUOT
METRICS_ACR_AROPOT	The cache table acronym "METRICS_ACR_AROPOT" must be short and thus it stands for "All Custom Reports Active Resources Over Period Of Time". It is cached once a day from the real-time query "ACR_ActiveResourcesOverPeriodOfTime". It is access from AllCustomReports.ActiveResourcesOverPeriodOfTime. TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_AllCustom_ActiveResourcesOverPeriodOfTime → READ: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ ACR_ActiveResourcesOverPeriodOfTime → INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_ACR_AROPOT
METRICS_ACR_RCD	The cache table acronym "METRICS_ACR_RCD" must be short and thus it stands for "All Custom Reports Resource Count Details". It is cached once a day from the real-time query "ACR_ResourceCount_Details". It is access from AllCustomReports.ResourceCount_Details. TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_21_AllCustom_ResourceCount_Details → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_AllCustom_ResourceCount_Details → READ: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ACR_ResourceCountDetails → INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_ACR_RCD
METRICS_ACR_RCT	The cache table acronym "METRICS_ACR_RCT" must be short and thus it stands for "All Custom Reports Resource Count Total". It is cached once a day from the real-time query "ACR_ResourceCount_Total". It is access from AllCustomReports.ResourceCount_Total. TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_22_AllCustom_ResourceCount_Total → PROCEDURE:

© Copyright 2000-2017 TIBCO Software Inc.

	/KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_AllCustom_ResourceCount_Total→ READ: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ACR_ResourceCount_Total → INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_ACR_RCT
METRICS_ALL_RESOURCES	This table stores a cached representation of the DV system ALL_RESOURCES table because it is more efficient to query. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_01_Cache_ALL_RESOURCES → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_RESOURCES → READ: /KPImetrics/Physical/Metadata/System/ALL_RESOURCES → INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_ALL_RESOURCES
METRICS_ALL_USERS	This table stores user information for all user accounts that have executed queries against the DV instance. This is an incremental cache target table. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_02_Cache_ALL_USERS → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_USERS → READ: /KPImetrics/Physical/Metadata/System/ALL_USERS → INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_ALL_USERS
METRICS_CIS_SYSTEM_RESOURCES	This tables stores hourly snapshots of memory, disk and I/O usage captured from SYS_MEMORY, SYS_STORAGE and SYS_IO tables. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_CIS_SYSTEM_RESOURCES→ READ: /KPImetrics/Physical/Physical/Abstraction/METRICS_CIS_SYSTEM_RESOURCES + /KPImetrics/Physical/Metadata/System/Helpers/p15MinutesIncrements INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_CIS_SYSTEM_RESOURCES
METRICS_CIS_WORKFLOW	The CIS_WORKFLOW table manages workflows for processing data. PROCEDURE: /KPImetrics/Physical/Metadata/System/Helpers/pStartWorkflow /KPImetrics/Physical/Metadata/System/Helpers/pEndWorkflow

© Copyright 2000-2017 TIBCO Software Inc.

	INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_CIS_WORKFLOW
METRICS_CPU_MEMORY_CHECKER	This table stores the results of executing the CPUAndMemChecker procedure over time. This is an incremental cache target table. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_CPU_MEMORY_CHECKER→ READ: /KPImetrics/Physical/Metadata/System/CPU_MEMORY_CHECKER → /KPImetrics/Physical/Metadata/CPUAndMemChecker/CpuAndMemCheckerCjp → CPU Utilization KPImetricsCpuUtilization.ps1 KPImetricsTopCommandGrepCpu_linux6.sh KPImetricsTopCommandGrepCpu_linux7.sh Memory Utilization KPImetricsMemUtilization.ps1 KPImetricsFreeMemCommand_linux6.sh KPImetricsFreeMemCommand_linux7.sh INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_CPU_MEMORY_CHECKER
METRICS_EVENT_REG_LOG	This table stores the details about the events that have occurred and the emails that have been sent out. An event is only logged if an email is sent. Events are registered in the METRICS_EVENT_REGISRATION table and include: LONG_RUNNING, EXCEEDED_MEMORY, INACTIVITY and PURGE_HISTORY. The event time, the user, actual email along with the SQL description when applicable is stored. TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ 1. kpimetricsTrig_15_CheckMetricsActivity → Event=INACTIVITY 2. kpimetricsTrig_16_PurgeHistoryData → Event=PURGE_HISTORY 3. kpimetricsTrig_17_CheckExceedMemoryPercentRequests → Event=EXCEEDED_MEMORY 4. kpimetricsTrig_18_CheckLongRunningRequests →Event=LONG_RUNNING PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ 1. pCheckMetricsActivity → ../Helpers/pUpdateEventRegLog 2. pPurgeData → ../Helpers/pUpdateEventRegLog 3. pCheckExceededMemoryPercentRequests → ../Helpers/pUpdateEventRegLog 4. pCheckLongRunningRequests → ../Helpers/pUpdateEventRegLog READ: /shared/ASAssets/KPImetrics/Physical/Metadata/System/ 1. METRICS_JOB_DETAILS 2. N/A 3. /KPImetrics/Business/Business/requests/vExceededMemoryPercentRequests 4. /KPImetrics/Business/Business/requests/vLongRunningRequests INSERT:

© Copyright 2000-2017 TIBCO Software Inc.

	/KPImetrics/Physical/Physical/Abstraction/METRICS_EVENT_REG_LOG and METRICS_EVENT_REG_LOG_LINEAGE
METRICS_EVENT_REG_LOG_LINEAGE	This table is a child table to METRICS_EVENT_REG_LOG and is used to store the data source lineage for the SQL request that was logged as a result of either LONG_RUNNING or EXCEEDED_MEMORY. The other events do not produce a SQL description so are not applicable. A SQL description is parsed using the same procedures that produce the METRICS_SQL_REQUEST data and store one or more rows associated with the requests data source and connection information. The reporting on this can look for events and quickly determine which data source is being accessed which can assist the viewer on how to take action with the data owners if necessary.
METRICS_EVENT_REGISTRATION	This table is used to register a metrics event registration. A subscription in the METRICS_EVENT_REGISTRATION table consists of a unique record for the combination of SUBSCRIBER_EMAIL, GROUP_NAME, ENVIRONMENT_TYPE, EVENT_TYPE and REQUESTER_EMAIL. The EVENT_TYPE can be one of [LONG_RUNNING| EXCEEDED_MEMORY| INACTIVITY| WORKFLOW_FAILURE| DBMS_SCHEDULER_ERROR].
METRICS_JOB_DETAILS	This table is used to hold the data transfer job details when rows are moved from the metrics collection tables to the history tables. The native database procedure "P_METRICS_ALL_TABLES" performs the following data transfer capabilities: metrics_sessions → metrics_sessions_hist metrics_resources_usage → metrics_resources_usage_hist metrics_requests → metrics_requests_hist
METRICS_JOB_DELETE_CHECK	This table provides information about TDV metrics delete statement activity when trigger "kpimetricsTrig_35_DBMSDeleteCheck" is turned on. It will reveal when the delete statement was executed for each node and how many rows were deleted from the collection tables.
METRICS_JOB_ENVIRONMENTS	This table provides a list of valid environments. In essence, the ENV_TYPE is like the short nickname for a host. For example, DEV1 is the short-name the development server. This is used by various email notification procedures.
METRICS_JOB_FILTERS	This table is used to hold the job filters used by "P_METRICS_ALL_TABLES" whereby the metrics_resource_usage collection table filters out rows based on user, domain and resourcekind. This capability allows rows to be filtered out before they get to the history table thus reducing the overall burden. Without this feature, the database would be overwhelmed by millions of unnecessary rows as absolutely everything in DV is reported.
METRICS_LDAP_PERSON	This LDAP_PERSON table is used to pre-cache LDAP user information. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_03_Cache_LDAP_PERSON→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LDAP_PERSON→ READ: /shared/ASAssets/KPImetrics/Configuration/Customize/LDAP_PERSON INSERT: synchronous cache refresh executed on one of applicable tables: /KPImetrics/Physical/Physical/KPI_oracle/METRICS_LDAP_PERSON /KPImetrics/Physical/Physical/KPI_sqlserver/METRICS_LDAP_PERSON
METRICS_LOG_DISK	This table stores logs of available disk space incrementally cached from the DV system table LOG_DISK. It is processed using the system interface lineage: TRIGGER:

© Copyright 2000-2017 TIBCO Software Inc.

	/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_06_Cache_LOG_DISK→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_DISK→ READ: /KPImetrics/Physical/Metadata/System/LOG_DISK INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_LOG_DISK
METRICS_LOG_IO	This table stores IO logs incrementally cached from the DV system table LOG_IO. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_07_Cache_LOG_IO→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_IO→ READ: /KPImetrics/Physical/Metadata/System/LOG_IO INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_LOG_IO
METRICS_LOG_MEMORY	This table stores jvm memory logs incrementally cached from the DV system table LOG_MEMORY. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_08_Cache_LOG_MEMORY → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_MEMORY → READ: /KPImetrics/Physical/Metadata/System/LOG_MEMORY INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_LOG_MEMORY
metrics_requests	This is the DV out-of-the-box requests table. It cannot contain any indexes. Its purpose is simply to be a collector of DV metrics requests.
metrics_requests_hist	This is the KPImetrics historical requests table. It may be partitioned and contain indexes for better query performance. It is updated periodically from the metrics_requests table. If partitioned by month, it allows a more efficient way to purge data by dropping a partition rather than deleting data.
metrics_requests_stg	This is a KPImetrics stage tables used to perform updates and deletes on metrics_requests collection data before inserting into metrics_requests_hist.
metrics_requests_stg_upd	This is a KPImetrics historical stage table that is used during the data transfer procedure for doing a mass update on the metrics_requests_stg table with Oracle.
metrics_resources_usage	This is the DV out-of-the-box resources usage table. It cannot contain any indexes. Its purpose is simply to be a collector of DV metrics resources usage.
metrics_resources_usage_stg	This is a KPImetrics stage tables used to perform updates on metrics_resources_usage collection data before inserting into metrics_resources_usage_hist.
metrics_resources_usage_hist	This is the KPImetrics historical resources usage table. It may be partitioned and contain indexes for better query performance. It is updated periodically from the

© Copyright 2000-2017 TIBCO Software Inc.

	metrics_resources_usage table. If partitioned by month, it allows a more efficient way to purge data by dropping a partition rather than deleting data.
metrics_sessions	This is the DV out-of-the-box sessions table. It cannot contain any indexes. Its purpose is simply to be a collector of DV metrics sessions.
metrics_sessions_stg	This is a KPImetrics stage tables used to perform updates on metrics_sessions collection data before inserting into metrics_sessions_hist.
metrics_sessions_hist	This is the KPImetrics historical sessions table. It may be partitioned and contain indexes for better query performance. It is updated periodically from the metrics_sessions table. If partitioned by month, it allows a more efficient way to purge data by dropping a partition rather than deleting data.
METRICS_SQL_COLUMNS	This table holds the parsed SQL columns for the query statement. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST and kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_METRICS_SQL_REQUEST_EXEC → Cache_METRICS_SQL_REQUEST_GENSQL and Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS→ Cache_METRICS_SQL_REQUEST_GENSQL_REPROCESS READ: /KPImetrics/Physical/Formatting/metrics_requests_hist + /KPImetrics/Physical/Formatting/METRICS_SQL_REQUEST INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_SQL_COLUMNS
METRICS_SQL_REQUEST	This table holds the parsed SQL status for the query statement. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST and kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_METRICS_SQL_REQUEST_EXEC → Cache_METRICS_SQL_REQUEST_GENSQL and Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS→ Cache_METRICS_SQL_REQUEST_GENSQL_REPROCESS READ: /KPImetrics/Physical/Formatting/metrics_requests_hist + /KPImetrics/Physical/Formatting/METRICS_SQL_REQUEST INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_SQL_REQUEST
METRICS_SQL_CONTROL	GATEKEEPER - SEMAPHORE: Used by Cache_METRICS_SQL_REQUEST_EXEC. This section of code acts a semaphore to determine whether to process records or not. This procedure is allowed to process the records that were generated from a different node in order to more efficiently share in the workload. This is important as some nodes in a

© Copyright 2000-2017 TIBCO Software Inc.

	cluster may generate more rows than others due to an imbalance in the load balancer due to "sticky" setting instead of a true "round-robin" setting. Upon entering this procedure, if there are rows in the METRICS_SQL_REQUEST table where KPI_PROCESS_TIME is null and the PROCESSED_NODE_HOST and PROCESSED_NODE_PORT matches this nodes nodehost and nodeport, then it must complete its current work before starting any new work. This procedure can only execute the pre-processing by one node at a time within a cluster because this procedure will work on data from other nodes if the current node has no more work to do. It is imperative that each node be allowed time to pre-insert the number of rows identified by the variable "numRowsToProcessBeforeExiting" into the METRICS_SQL_REQUEST table. For example, 500 rows would be pre-inserted with the KPI_PROCESS_TIME being set to null. The acutal processing will take place based on those rows. As soon as the pre-processing has completed, the control record will be removed from the METRICS_SQL_CONTROL table thus allowing another node to perform its pre-processing.
METRICS_SQL_CONTROL_LOG	Contains a log of the Gatekeeper code block within Cache_METRICS_SQL_REQUEST_EXEC when debugGatekeeper = '1'. During normal operation, this will be turned off so that no rows are produced. It is only useful for debugging purposes to ensure that in a clustered environment, the nodes are taking their turn initializing their own set of rows when doing parallel processing on one of the node's data.
METRICS_SQL_RESOURCE	This table holds the parsed SQL resource for the query statement. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST and kpimetricsTrig_12_Cache_METRICS_SQL_REQUEST_REPROCESS→ PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_METRICS_SQL_REQUEST_EXEC → Cache_METRICS_SQL_REQUEST_GENSQL and Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS→ Cache_METRICS_SQL_REQUEST_GENSQL_REPROCESS READ: /KPImetrics/Physical/Formatting/metrics_requests_hist + /KPImetrics/Physical/Formatting/METRICS_SQL_REQUEST INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_SQL_RESOURCE
METRICS_SQL_RESOURCE_LINEAGE	This table holds the parsed SQL data source resource for each resource in the query statement. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/ kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/ Cache_METRICS_SQL_RESOURCE_LINEAGE READ: /KPImetrics/Physical/Physical/Abstraction/METRICS_SQL_RESOURCE + /KPImetrics/Physical/Physical/Abstraction/METRICS_SQL_RESOURCE_LINEAGE

© Copyright 2000-2017 TIBCO Software Inc.

	INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_SQL_RESOURCE_LINEAGE
METRICS_SYS_CACHES	This table stores a list of all cached resources and their current statuses incrementally cached from the DV system table SYS_CACHES. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_12_Cache_SYS_CACHES → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_CACHES → READ: /KPImetrics/Physical/Metadata/System/SYS_CACHES INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_SYS_CACHES
METRICS_SYS_DATASOURCES	This table stores a list of all data sources and their current statuses incrementally cached from the DV system table SYS_DATASOURCES. It is processed using the system interface lineage: TRIGGER: /KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_13_Cache_SYS_DATASOURCES → PROCEDURE: /KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_DATASOURCES → READ: /KPImetrics/Physical/Metadata/System/SYS_DATASOURCES INSERT: /KPImetrics/Physical/Physical/Abstraction/METRICS_SYS_DATASOURCES

© Copyright 2000-2017 TIBCO Software Inc.

Metadata System Triggers and Load Scripts

Location: /shared/ASAssets/KPImetrics/Physical/Metadata/System

/ClusterSafeCache

/ClusterSafeTriggers

/Helpers

This section provides a quick summary of all triggers, their schedules and how they execute in a cluster.

Note: "all nodes" and cluster dedicated timekeeper…

The reference to "**all nodes**" refers to all working nodes in a cluster except if there is a dedicated timekeeper. If there is no dedicated timekeeper then one of the nodes is nominated to be a timekeeper. KPImetrics will execute on that node.

When there is a dedicated timekeeper, then KPImetrics procedures will not execute on those nodes as configured in commonValues.dedicatedTimeKeeperHostname and commonValues.dedicatedTimeKeeperPort.

For "only once per cluster", whichever node is the timekeeper nominates a single node in the cluster to perform the work.

Trigger Name	Trigger Schedule	Trigger Period	Cluster execution
kpimetricsTrig_00_CheckMetricsActivityDebug [OFF]	1:00 AM	1 hour	all nodes
kpimetricsTrig_01_Cache_ALL_RESOURCES	1:00 AM	2 hours	all nodes
kpimetricsTrig_02_Cache_ALL_USERS	1:00 AM	2 hours	all nodes
kpimetricsTrig_03_Cache_LDAP_PERSON	6:00 AM	1 day	only once per cluster
kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES	1:15 AM	1 hour	all nodes
kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER	1:00 AM	1 hour	all nodes
kpimetricsTrig_06_Cache_LOG_DISK	1:00 AM	1 hour	all nodes
kpimetricsTrig_07_Cache_LOG_IO	1:00 AM	1 hour	all nodes
kpimetricsTrig_08_Cache_LOG_MEMORY	1:00 AM	1 hour	all nodes
kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST [OFF]	1:15 AM	30 min	all nodes
kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS [OFF]	1:00 AM	1 day	all nodes
kpimetricsTrig_12_Cache_SYS_CACHES	1:00 AM	6 hours	all nodes
kpimetricsTrig_13_Cache_SYS_DATASOURCES	6:00 AM	12 hours	all nodes
kpimetricsTrig_14_CheckCISWorkflowStatusFail	1:30 AM	1 hour	all nodes
kpimetricsTrig_15_CheckMetricsActivity	12:59:50 AM	1 day	all nodes
kpimetricsTrig_16_PurgeHistoryData	4:30 AM	1 week	only once per cluster
kpimetricsTrig_17_CheckExceedMemoryPercentRequests	1:00 AM	1 hour	all nodes
kpimetricsTrig_18_CheckLongRunningRequests	system event	RequestRunForTooLong	all nodes
kpimetricsTrig_19_AllCustom_AccessByUserOverTime	12:15 AM	1 day	all nodes
kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime	12:15 AM	1 day	all nodes
kpimetricsTrig_21_AllCustom_ResourceCount_Details	12:15 AM	1 day	all nodes
kpimetricsTrig_22_AllCustom_ResourceCount_Total	12:15 AM	1 day	all nodes

© Copyright 2000-2017 TIBCO Software Inc.

kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE [OFF]	1:10 AM	1 hour	all nodes
kpimetricsTrig_24_ValidateUpdateStatusTables	1:10 AM	30 min	all nodes
kpimetricsTrig_25_ExecuteMetricsSqlControl	1:05 AM	15 min	all nodes
kpimetricsTrig_30_DBMSScheduler	12:30 AM	2 hours	only once per cluster
kpimetricsTrig_31_DBMSSchedulerError	1:00 AM	2 hours	only once per cluster
kpimetricsTrig_32_DBMSPartitionManager	12:00 AM	1 day	only once per cluster
kpimetricsTrig_33_DeleteCollection [OFF] [DEPRECATED]	1:05 AM	30 min	only once per cluster
kpimetricsTrig_34_DBMSRebuildIndexes	3:00 AM	1 week [Sunday]	only once per cluster
kpimetricsTrig_40_Cache_METADATA_TABLES [OFF]	9:30 PM	1 day	all nodes
kpimetricsTrig_50_MetricsPersistenceFailure	system event	MetricsPersistentFailure	all nodes
kpimetricsTrig_51_MetricsTruncateFailure	system event	MetricsTruncationFailure	all nodes
kpimetricsTrig_52_MetricsBackupFailure	system event	MetricsBackupFailure	all nodes
kpimetricsTrig_53_MetricsRestoreFailure	system event	MetricsRestoreFailure	all nodes

This section lists all triggers and load scripts that have been defined to execute various KPImetrics procedures at regular intervals. The default execution frequencies are listed for each trigger. The load scripts have been created to load and aggregate raw data into processed KPImetrics metrics.

Recommendation: Set the following Administration → Configuration parameter

Path	Type	Description	Value
/server/config/cluster/triggerDistribution/timeKeeperWeight	INTEGER	Weight of time keeper	0

Trigger [schedule] → Script Name → View name	Description
Schedule: [1 hour, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_00_CheckMetricsActivityDebug 'Y',60,06:00:00,20:00:00 → pCheckMetricsActivity →** pGetEmailSubscriptions	DEFAULT=OFF This script is used for debugging the DV native metrics and checks for activity on an hourly basis and alerts the configured user whether there was any inactivity. This can be a useful gauge of the health of the DV system. It is configured hourly, but it may make sense to alter the timing to every 2 or more hours depending activity-levels. It is also configured to check between the hours of 6 am to 8 pm daily. It will look at the current timestamp and compare with MAX(starttime) for the collection tables [metrics_requests, metrics_resources_usage] and MAX(logintime) for [metrics_sessions]. It also looks to see if the data transfer jobs have been running by looking at METRICS_JOB_DETAILS. Email integration must be configured on the DV server for this to work.
Schedule: [2hours, 1:00 am] **[Executes on all nodes in the cluster]**	Cache ALL_RESOURCES to make joining with other KPImetrics tables more efficient. This is not a

© Copyright 2000-2017 TIBCO Software Inc.

kpimetricsTrig_01_Cache_ALL_RESOURCES → **Cache_ALL_RESOURCES →** /System/ALL_RESOURCES → [/services/databases/system/ALL_RESOURCES, l_hostname, l_port] → METRICS_ALL_RESOURCES	historical view of resources. It gets a new copy each time it executes and deletes the old copy. This procedure will first check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_RESOURCES%' and delete those records before executing. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. Insert into METRICS_ALL_RESOURCES select from /System/ALL_RESOURCES
Schedule: [2 hours, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_02_Cache_ALL_USERS →** **Cache_ALL_USERS →** /System/ALL_USERS → [/services/databases/system/ALL_USERS, l_hostname, l_port] → METRICS_ALL_USERS	Cache ALL_USERS to make joining with other KPImetrics tables more efficient. This is not a historical view of users. It gets a new copy each time it executes and deletes the old copy. This procedure will first check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_USERS%' and delete those records before executing. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. Insert into METRICS_ALL_USERS select from /System/All_USERS
Schedule: [1 day, 6:00 am] **[Only once per cluster]** **kpimetricsTrig_03_Cache_LDAP_PERSON →** **Cache_LDAP_PERSON →** ./ /shared/ASAssets/KPImetrics/Configuration/Customize/LDAP_PERSON→ /KPImetrics/Physical/Metadata/LDAP/organizationalPerson	Cache LDAP person data once a day. This is not a historical view of users. It gets a new copy each day. Insert into METRICS_LDAP_PERSON select from ./Metadata/System/LDAP_PERSON
Schedule: [1 hour, 1:15 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES →** **Cache_CIS_SYSTEM_RESOURCES →** p15MinutesIncrements	Insert 15 min increment rows into METRICS_CIS_SYSTEM_RESOURCES select from METRICS_LOG_MEMORY, METRICS_LOG_IO, METRICS_LOG_DISK, METRICS_CPU_MEMORY_CHECKER.
Schedule: [1 hour, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER →** **Cache_CPU_MEMORY_CHECKER →** ./Metadata/System/CPU_MEMORY_CHECKER → /KPImetrics/Physical/Metadata/ CPUAndMemChecker/CpuAndMemCheckerCjp	Cache the system CPU and memory values. Insert into METRICS_CPU_MEMORY_CHECKER select from ./Metadata/System/CPU_MEMORY_CHECKER.
Schedule: [1 hour, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_06_Cache_LOG_DISK →** **Cache_LOG_DISK→** ./Metadata/System/LOG_DISK → /services/databases/system/LOG_DISK	Cache DV system log disk information. Insert into METRICS_LOG_DISK select from ./Metadata/System/LOG_DISK

© Copyright 2000-2017 TIBCO Software Inc.

Schedule: [1 hour, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_07_Cache_LOG_IO [→ Cache_LOG_IO →** ./Metadata/System/LOG_IO → /services/databases/system/LOG_IO	Cache DV system IO information. Insert into METRICS_LOG_IO select from ./Metadata/System/LOG_IO
Schedule: [1 hour, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_08_Cache_LOG_MEMORY → Cache_LOG_MEMORY →** ./Metadata/System/LOG_MEMORY → /services/databases/system/LOG_MEMORY	Cache DV system log memory information. Insert into METRICS_LOG_MEMORY select from ./Metadata/System/LOG_MEMORY
Schedule: [30 min, 1:15 am] – Primary configuration. **Schedule: [15 min, 1:15 am] – Alternative if more frequent processing is needed.** **[Executes on all nodes in the cluster]** **kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST → Cache_METRICS_SQL_REQUEST_EXEC →** Cache_METRICS_SQL_REQUEST_GENSQL	DEFAULT=OFF Execute the SQL Parser to parse the SQL request (description) originating from metrics_requests_hist. Retrieve the list of tables and colunms and update the METRICS_SQL_REQUEST, METRICS_SQL_RESOURCE and METRICS_SQL_COLUMNS tables. This procedure will first check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_RESOURCES%' and delete those records before executing Cache_ALL_RESOURCES. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. This procedure is dependent on METRICS_ALL_RESOURCES data. The schedule is designed to run 15 min after **kpimetricsTrig_30_DBMSScheduler** allowing that trigger to execute pMETRICS_ALL_TABLES_exec which updates the metrics_requests_hist table. The data from metrics_requests_hist is what feeds this trigger and procedure. This procedure runs every 30 min because it often runs behind in its work due to the intensity level at which runs. This trigger blocks (does not run) if Cache_METRICS_SQL_RESOURCE_LINEAGE is currently executing. The two procedures running together causes DV and the database to become overwhelmed. This procedure is architected for multi-host processing. If there are no rows to process for the current node, it will look to see which node (if cluster) has the most rows to process and help process those rows. This code uses a semaphore to reserve rows for processing. The row reservation can only be executed by one node at a time. The actual processing is done in parallel.

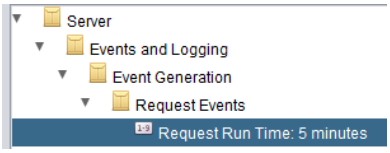
© Copyright 2000-2017 TIBCO Software Inc.

	NOTE: It may be necessary to set the trigger to 15 min if more frequent processing is required and it can still keep up with the number rows to process in the time period. The number of rows can be found in commonValues.numRowsToProcessBeforeExiting. Adjust the default value of 500 to whatever makes sense for each environment. When running in a cluster, this trigger/procedure will attempt to process another nodes SQL once it has processed all of its own. This parallel processing is important as some nodes may be slower than others and fall behind or the load balancer may be set to "sticky" instead of a true "round-robin" and the majority of requests end up on a few of the nodes in the cluster.
[No trigger – manual] Cache_METRICS_SQL_REQUEST_EXEC_ADHOC → Cache_METRICS_SQL_REQUEST_GENSQL_ADHOC	Provides a way to manually re-process successfully parsed requests. This is only necessary if a code-patch has been provided that changes the values that are inserted. Normally, this will never be executed.
Schedule: [1 day, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS → Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS** → Cache_METRICS_SQL_REQUEST_GENSQL_REPROCESS (0)	DEFAULT=OFF Re-process errors and execute the SQL Parser to parse the SQL request (description) originating from metrics_requests_hist. Retrieve the list of tables and colunms and update the METRICS_SQL_REQUEST, METRICS_SQL_RESOURCE and METRICS_SQL_COLUMNS tables. Typically, this trigger is not turned on unless deemed necessary. This procedure will first check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_RESOURCES%' and delete those records before executing Cache_ALL_RESOURCES. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. This procedure is dependent on METRICS_ALL_RESOURCES data. This trigger blocks (does not run) if Cache_METRICS_SQL_RESOURCE_LINEAGE is currently executing. The two procedures running together causes DV and the database to become overwhelmed.
Schedule: [6 hours, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_12_Cache_SYS_CACHES → Cache_SYS_CACHES** → ./Metadata/System/SYS_CACHES → /services/databases/system/SYS_CACHES	Cache the system cache status information. Insert into METRICS_SYS_CACHES select from / ./Metadata/System/SYS_CACHES.
Schedule: [12 hours, 6:00 am]	Cache the system data source information.

© Copyright 2000-2017 TIBCO Software Inc.

[Executes on all nodes in the cluster] **kpimetricsTrig_13_Cache_SYS_DATASOURCES [→ Cache_SYS_DATASOURCES →** ./Metadata/System/SYS_DATASOURCES → /services/databases/system/SYS_DATASOURCES	Insert into METRICS_SYS_DATASOURCES select from ./Metadata/System/SYS_DATASOURCES. Decide whether you want to check /users folders or not. The recommendation is to leave the value as 'N' to not check datasources in the /users folders. IN includeUsersFolders CHAR(1), - DEFAULT: N or null. Do not include /users folders. Y=include the datasources in /users folders.
Schedule: [1 hour, 1:30 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_14_CheckCISWorkflowStatusFail [→ pCheckCISWorkflowStatusFail →** [METRICS_CIS_WORKFLOW, pGetEmailSubscriptions]	Check for WORKFLOW_STATUS=F in the METRICS_CIS_WORKFLOW table since the last check. Each time this procedure is called it puts a marker row in the table with WORKFLOW_NAME=CHECK_WORKFLOW_STATUS Email integration must be configured on the DV server for this to work.
Schedule: [1 day, 12:59:50 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_15_CheckMetricsActivity 'N',60,00:00:00.000,23:59:59.999 → pCheckMetricsActivity** → pGetEmailSubscriptions	This script checks for activity on a daily basis just before midnight and alerts the configured user whether there was any inactivity during the day. It is configured to look for inactivity between the hours of 12 am – 12 pm in the current day. This can be a useful gauge of the health of the DV system. It will look at the current timestamp and compare with MAX(starttime) for the collection tables [metrics_requests, metrics_resources_usage] and MAX(logintime) for [metrics_sessions]. It also looks to see if the data transfer jobs have been running by looking at METRICS_JOB_DETAILS. Email integration must be configured on the DV server for this to work.
Schedule: [1 day, 4:30 am] **[Only once per cluster]** **kpimetricsTrig_16_PurgeHistoryData →** pPurgeData	This script purges old data from METRICS tables by executing series of DELETE statements. The purge period for each delete is defined within /Customize/commonValues script. purgeWorkflowData 120 = 4 months - Purge tables: METRICS_CIS_WORKFLOW purgeSQLRequests 120 = 4 months - Purge tables: METRICS_SQL_COLUMNS, METRICS_SQL_RESOURCE, METRICS_SQL_REQUEST purgeResourceUsage 120 = 4 months - Purge tables: METRICS_CIS_SYSTEM_RESOURCES, METRICS_CPU_MEMORY_CHECKER, METRICS_LOG_DISK, METRICS_LOG_IO, METRICS_LOG_MEMORY, METRICS_SYS_DATASOURCES Email integration must be configured on the DV server for this to work.
Schedule: [1 hour, 1:00 am] **[Executes on all nodes in the cluster]**	This procedure queries this procedure in real-time /shared/ASAssets/KPImetrics/Business/Business/reques

© Copyright 2000-2017 TIBCO Software Inc.

kpimetricsTrig_17_CheckExceedMemoryPercentRequests → **pCheckExceedMemoryPercentRequests →** pGetEmailSubscriptions	ts/pExceededMemoryPercentRequests to generate a list of queries exceeding memory percent per request. It generates an html table containing each of the requests and emails to the subscriber of the event [EXCEEDED_MEMORY]. Email integration must be configured on the DV server for this to work.
Schedule: [system event=requestRunForTooLong] **[Executes on all nodes in the cluster]** **kpimetricsTrig_18_CheckLongRunningRequests →** **pCheckLongRunningRequests →** pGetEmailSubscriptions	This procedure queries this procedure in real-time /shared/ASAssets/KPImetrics/Business/requests/ pLongRunningRequests to generate a list of long running requests. It generates an html table containing each of the requests and emails to the subscriber of the event [LONG_RUNNING]. Email integration must be configured on the DV server for this to work. The trigger is activated by the system request event: "Request Run Time". This is set in the Administration Configuration. 
Schedule: [1 day, 12:15 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_19_AllCustom_AccessByUserOvertime →** **Cache_AllCustom_AccessByUserOvertime →** /Metadata/Physical/Abstraction/EXCEPT_ACR_AccessByUser Overtime → [metrics_resources_usage_hist NOT EXISTS METRICS_ACR_ABUOT]	This procedure ivokes "ACR_AccessByUserOvertime" once a day to improve overall query performance for this report. It finds the delta/difference in rows using the original underlying view metrics_resources_usage_hist and NOT EXISTS in METRICS_ACR_ABUOT. The acronym for the cache table must be short and thus it stands for "All Custom Reports Access By User Over Time".
Schedule: [1 day, 12:15 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOf Time →** **Cache_AllCustom_ActiveResourcesOverPeriodOfTime→** /Metadata/Physical/Abstraction/EXCEPT_ACR_ActiveResourc esOverPeriodOfTime → [metrics_resources_usage_hist NOT EXISTS METRICS_ACR_ABUOT]	This procedure invokes "EXCEPT_ACR_ActiveResourcesOverPeriodOfTime" once a day to improve overall query performance for this report. It finds the delta/difference in rows using the original underlying view metrics_resources_usage_hist and NOT EXISTS in METRICS_ACR_AROPOT. The acronym for the cache table must be short and thus it stands for "All Custom Reports Active Resources Over Period Of Time".
Schedule: [1 day, 12:15 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_21_AllCustom_ResourceCount_Details →** **Cache_AllCustom_ResourceCount_Details→** /AllCustomReports/ResourceCount_DetailsRT → vResourceUsageUD → metrics_resources_usage_hist	This procedure caches the real-time query "ACR_ResourceCount_DetailsRT" once a day to improve overall query performance for this report. It caches a completely new set of rows to METRICS_ACR_RCD. The acronym for the cache table must be short and thus it stands for "All Custom Reports Resource Count Details".
Schedule: [1 day, 12:15 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_22_AllCustom_ResourceCount_Total →** **Cache_AllCustom_ResourceCount_Total→**	This procedure caches the real-time query "ACR_ResourceCount_TotalRT" once a day to improve overall query performance for this report. It caches a completely new set of rows to METRICS_ACR_RCT. The acronym for the cache

© Copyright 2000-2017 TIBCO Software Inc.

/AllCustomReports/ResourceCount_TotalRT→ vResourceUsageUD → metrics_resources_usage_hist	table must be short and thus it stands for "All Custom Reports Resource Count Total".
Schedule: [1 hour, 1:10 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE** → Cache_METRICS_SQL_RESOURCE_LINEAGE → METRICS_SQL_RESOURCE and METRICS_SQL_RESOURCE_LINEAGE	DEFAULT=OFF This procedure will insert the new data into METRICS_SQL_RESOURCE_LINEAGE based on the START_TIME value from the table METRICS_SQL_RESOURCE. The lineage is derived by either copying a similar resource from the database table or by generating the lineage using getResourceLineageDirectRecursive(). This procedure will first check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_RESOURCES%' and delete those records before executing Cache_ALL_RESOURCES. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. This procedure is dependent on METRICS_ALL_RESOURCES data. This procedure will block and not execute if Cache_METRICS_SQL_REQUEST_[EXEC or_ADHOC or _REPROCESS] is running. The reason is that the two procedures combined execution can overwhelm both the database and the DV node. This procedure is architected for multi-host processing. If there are no rows to process for the current node, it will look to see which node (if cluster) has the most rows to process and help process those rows. This code uses a semaphore to reserve rows for processing. The row reservation can only be executed by one node at a time. The actual processing is done in parallel. The lineage is generated for each non-null RESOURCE_PATH and RESOURCE_TYPE found in the METRICS_SQL_RESOURCE table. METRICS_SQL_REQUEST [parent] --> [0 to many] METRICS_SQL_RESOURCE [child] --> [0 to many] METRICS_SQL_RESOURCE_LINEAGE [child] 1. A request may have 0 or more resources. 2. A resource may have 0 or more data source resources. a. A single resource lineage row contains the Ancestor who uses the data source resource, the data source table/procedure/tree resource and the data source itself.
Schedule: [30 min, 1:10 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_24_ValidateUpdateStatusTables →** pValidateUpdateStatusTables	This procedure is used to validate the two status tables and update any failed "in progress" records that are no longer actively running in the TDV memory space as per system.SYS_REQUESTS table.

© Copyright 2000-2017 TIBCO Software Inc.

	This procedure works in conjunction with "pMETRICS_ALL_TABLES_exec" and "pDELETE_COLLECTION_exec" to check for active running processes. This procedure is used to ensure that the METRICS_CIS_WORKFLOW and METRICS_JOB_DETAILS do not have in progress status when there are no active processes. It is the responsibility of each node in a cluster to manage its own status records. • For METRICS_CIS_WORKFLOW where WORKFLOW_STATUS='I' and there are no active processes, update WORKFLOW_STATUS='F' and set MESSAGE='Failure for unknown reason.'. • For METRICS_JOB_DETAILS where STATUS='BEGIN JOB' and there are no active processes, update STATUS='FAILURE' and set ADDITIONAL_INFO='Failure for unknown reason.'.
Schedule: [15 min, 1:05 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_25_ExecuteMetricsSqlControl →** pExecuteMetricsSqlControl	This procedure compares the current nodehost/nodeport with what is present in METRICS_SQL_CONTROL and is used to control the execution of the procedure referenced in the CONTROL_NAME field. These rows were inserted by updateTriggers according to the following rules. Both Cache_ALL_RESOURCES and Cache_ALL_USERS are required to be executed on each node of the cluster for the dependent resources to be able to use the data collected. The procedure “pExecuteMetricsSqlControl” will execute on each node of the cluster and fulfill the duty for both Cache_ALL_RESOURCES and Cache_ALL_USERS. In the event that this trigger does not get executed before one of the dependents listed below, the dependent procedure will execute what it needs to fulfill its job. Rules: Cache_ALL_RESOURCES gets executed by the dependents: Cache_ALL_RESOURCES Cache_METRICS_SQL_REQUEST_EXEC Cache_METRICS_SQL_REQUEST_EXEC_ADHOC Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS Cache_METRICS_SQL_RESOURCE_LINEAGE pMETRICS_ALL_TABLES_exec Cache_ALL_USERS gets executed by the dependents: Cache_ALL_USERS pMETRICS_ALL_TABLES_exec
Schedule: [2 hours, 12:30 am] **[Only once per cluster]**	This trigger executes the pMETRICS_ALL_TABLES_exec PLSQL procedure to

© Copyright 2000-2017 TIBCO Software Inc.

kpimetricsTrig_30_DBMSScheduler → **pMETRICS_ALL_TABLES_exec →** P_METRICS_ALL_TABLES()	transfer data from the metrics collection tables to the metrics history tables. It inserts a record into the METRICS_JOB_DETAILS table when it starts with a JOB_TABLE_NAME='DBMS_SCHEDULER'. It updates the same row with a STATUS='SUCCESS' or 'FAILURE'. If 'FAILURE' then update the ADDITIONAL_INFO field with the database error. This procedure will first check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_RESOURCES%' and delete those records before executing Cache_ALL_RESOURCES. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. This procedure is dependent on METRICS_ALL_RESOURCES data. Subsequently, this procedure will check METRICS_SQL_CONTROL for any records where NODE_HOST = current nodehost and NODE_PORT = current nodeport and CONTROL_NAME like 'Cache_ALL_USERS%' and delete those records before executing Cache_ALL_USERS. The records were inserted into METRICS_SQL_CONTROL by updateTriggers. This procedure is dependent on METRICS_ALL_USERS data. If the **30-min timeout occurs** because processing could not commense an exception will be thrown and logged in the METRICS_JOB_DETAIL table. Operational insights: 1. This procedure executes in a single-threaded fashion on one node in a cluster and processes all the data for each node in the cluster sequentially. It does this in order to avoid "deadlock" contention when multiple nodes issue the same commands at the same time. 2. This procedure will <u>not</u> execute if P_METRICS_DELETE_COLLECTION or pRebuildIndexes is executing. However, it will loop every minute and check if it can execute. It will wait up to 30 min to allow the other processes to complete.
Schedule: [2 hours, 1:00 am] **[Only once per cluster]** **kpimetricsTrig_31_DBMSSchedulerError →** **pCheckDBMSSchedulerError →** pGetEmailSubscriptions	Send an email if there is a database PLSQL data transfer error that gets generated. Select details from /Abstraction/METRICS_JOB_DETAILS. The timing of 2 hours on the odd hour is based on the fact that the DBMS Scheduler trigger runs every 2 hours on the even hour. Therefore, this trigger runs an hour later to allow the PLSQL data transfer script to complete and post any issues or not. Note: This trigger only runs once per cluster because it finds all errors for all nodes if there is a cluster.

© Copyright 2000-2017 TIBCO Software Inc.

Schedule: [1 day, 12:00 am] **[Only once per cluster]** **kpimetricsTrig_32_DBMSPartitionManager →** **pPARTITION_MANAGER_exec →** Oracle /KPImetrics/Physical/Metadata/DDL/Oracle/ [03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_ADD, 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_DROP] SqlServer /KPImetrics/Physical/Metadata/DDL/SqlServer/ [03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_ADD, 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_DROP]	Partition management is required for the metrics history tables when commonValues.partitionNumber and partitionStartDate are configured. The partition manager trigger wakes up once a day at 12 am and determines if a partition needs to be added or dropped. Technically, the only time any actual action will take place is the 1st day of the month unless DV is down at 12 am on the 1st. This is why it is scheduled to run every day to address any downtime. For every day except the 1st, it will simply find no partitions to add or drop based on what is currently in place and how the parttionNumber is configured. For adding a partition, it always looks at the current Year/Month it executes in and calculates the partition for the next month and determines if it exists or not. For dropping a partition, it counts the current number+1 for next month and compares with the partitionNumber to determine if it should drop the oldest partition.
[DEPRECATED - TURNED OFF] **Schedule: [30 min, 1:05 am] – Primary configuration.** **Schedule: [15 min, 1:05 am] – Alternative if more frequent processing is needed.** **[Only once per cluster]** **kpimetricsTrig_33_DeleteCollection →** **pMETRICS_DELETE_COLLECTION_exec →** P_METRICS_DELETE_COLLECTION()	DEFAULT=OFF [DEPRECATED] As of the 2020Q100, this trigger is turned off to eliminate blocking which occurs from DV metrics issuing inserts/deletes at the same time as KPImetrics issuing delete statements. Goring forward, only DV metrics will manage inserts/deletes into the collection tables. This trigger executes the PLSQL procedure pMETRICS_DELETE_COLLECTION_exec to delete non-essential and unwanted rows from the two collection tables: metrics_resources_usage and metrics_requests. It inserts a record into the METRICS_JOB_DETAILS table when it starts with a JOB_TABLE_NAME='DELETE_COLLECTION'. It updates the same row with a STATUS='SUCCESS' or 'FAILURE'. If 'FAILURE' then update the ADDITIONAL_INFO field with the database error. This trigger executes only once per cluster node and processes all transactions for all hosts. The procedure "pMETRICS_DELETE_COLLECTION_exec()" provides the flow control for each node by getting the eligible members of the cluster and looping through each nodehost/nodeport combination until all work is done. If the **30-min timeout occurs** because processing could not commense an exception will be thrown and logged in the METRICS_JOB_DETAIL table. Exceptions: Emails will be sent if there are exceptions. It is recommended to have this turned on if the system is very active and producing 100K or more transactions in 1 hour or if the commonValues.queryRequestTemplate='Y' which will produce 100K or more transactions itself within 1 hour. This setting is used to keep the collection tables as

© Copyright 2000-2017 TIBCO Software Inc.

	small as possible of unwanted data based on METRICS_JOB_FILTERS rows. 1. metrics_resources_usage – delete from this collection table where a row matches METRICS_JOB_FILTERS based on user, domain and resourcekind. This delete gets executed first. 2. metrics_requests – delete from this collection table when a row does not exist in metrics_resources_usage collection table. This delete gets executed second. Operational insights: 1. This procedure executes in a single-threaded fashion on one node in a cluster and processes all the data for each node in the cluster sequentially. It does this in order to avoid "deadlock" contention when multiple nodes issue the same commands at the same time. 2. This procedure will <u>not</u> execute if P_METRICS_ALL_TABLES or pRebuildIndexes is executing. However, it will loop every minute and check if it can execute. It will wait up to 30 min to allow the other processes to complete. 3. If this procedure finds another record for the same nodehost and nodeport and JOB_TABLE_NAME='DELETE_COLLECTION' and STATUS='BEGIN JOB' then it determines that the record is stranded due to some unforeseen event and marks that record with a STATUS='FAILURE'. It then continues on as normal. 4. The timing of the kpimetricsTrig_33_DeleteCollection [DC] trigger is important. It is set to execute after the kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST [MSR] trigger and before the kpimetricsTrig_30_DBMSScheduler [DBS] trigger. It is recommended that the period be the same between kpimetricsTrig_33_DeleteCollection and kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST. If one gets changed from 30 to 15, then the other one should be changed also. The objective is to delete unwanted rows produced by the first trigger and delete as much as possible for DBMSScheduler runs so that it does not have to spend time querying through 100K of unwanted rows. Ultimately, DBMSSchduler to run as quickly as possible. The only way to do that is to keep the data set as small as possible. The chart below shows the time interaction between the three triggers that are responsible for the following: [MSR] – SQL parsing with a byproduct of creating metrics request volume, [DBS] – transferring data from collection to history and deleting collection. [DC] – deleting collection data with the objective of keeping collection tables small and manageable.

© Copyright 2000-2017 TIBCO Software Inc.

	30 min period on MSR and DC triggers: [MSR] [DBS] [DC] 12:00 12:05 ←del collection just before processing 12:30 12:30 12:35 15 min period on MSR and DC triggers: [MSR] [DBS] [DC] 12:00 12:05 12:15 12:20 ←del collection just before processing 12:30 12:30 12:35 12:45 12:50
Schedule: [1 week, 3:00 am] [Sunday] **[Only once per cluster]** **kpimetricsTrig_34_DBMSRebuildIndexes →** **pRebuildIndexesAdminInterface →** KPI_DV_AdminInterface/KPImetricsAdmin/AdminInterface/pRebuildIndexes → pRebuildIndexes	This trigger executes the pRebuildIndexesAdminInterface procedure. This procedure is used to rebuild or reorganize the indexes. The trigger must invoke this procedure so that it gets redirected through the admin interface. This allows the trigger to take on the personna of the KPI_DV_AdminInterface allowing it to execute code that requires a "user" to execute. A trigger cannot execute the code by itself. This procedure will turn off all triggers before proceeding and wait for the processes to finish as there can be no database activity while the index rebuild is in progress. When completed it will turn the triggers back on that were originally on when the process started. It is recommended to only execute once a week such as a Sunday at 2 am. Indexes do not start rebuilding until all processare are inactive. For METRICS_CIS_WORKFLOW, WORKFLOW_STATUS <> 'I'. For METRICS_JOB_DETAILS, STATUS <> 'BEGIN JOB'. If the **1-hour timeout occurs** an exception will be thrown and logged in the METRICS_JOB_DETAIL table. Exceptions: Emails will be sent if there are exceptions. Review the following view (table) for issues: /services/databases/ASAssets/KPImetrics/workflow/vJobDetails
Schedule: [1 Hour, 1:00 am] **[Executes on all nodes in the cluster]** **kpimetricsTrig_35_DBMSDeleteCheck →** **pMetricsDeleteCheck**	This trigger executes the **pMetricsDeleteCheck procedure.** This procedure logs the TDV metrics delete statements which are executed by DV metrics based on the schedule set in /policy/metrics. Whenever the /policy/metrics is enabled TDV uses that timestamp in the delete statement. It monitors SYS_REQUESTS for TDV metrics delete statements and inserts the information into METRICS_JOB_DELETE_CHECK. The view is published as workflow.vJobDeleteCheck and can be used to monitor when the delete statements occur and how many rows get deleted for each node.
Schedule: [1 day, 9:30 pm]	DEFAULT=OFF

© Copyright 2000-2017 TIBCO Software Inc.

[Executes on all nodes in the cluster] **kpimetricsTrig_40_Cache_METADATA_TABLES →** Cache_METADATA_TABLES	This procedure is used to cache all of the METADATA_* tables. Uses the same ALL_RESOURCE data from METRICS_ALL_RESOURCES_STG which gets cached every 2 hours. The data would be current as of 9 pm. This alleviates the need to recache data that was already cached. Therefore, there is a **dependency** on **Cache_ALL_RESOURCES** completing for a given node.
Schedule: [system event=MetricsPersistentFailure] **[Executes on all nodes in the cluster]** **kpimetricsTrig_50_MetricsPersistenceFailure →** pMetricsSystemEvent	DEFAULT=ON This is activated when the system event=MetricsPersistentFailure occurs. This procedure sends an email to the commonValues.sendTo list and the subscription list. If there is a subscription list then it also saves the record to the METRICS_EVENT_REG_LOG.
Schedule: [system event=MetricsTruncationFailure] **[Executes on all nodes in the cluster]** **kpimetricsTrig_51_MetricsTruncateFailure →** pMetricsSystemEvent	DEFAULT=ON This is activated when the system event=MetricsTruncationFailure occurs. This procedure sends an email to the commonValues.sendTo list and the subscription list. If there is a subscription list then it also saves the record to the METRICS_EVENT_REG_LOG.
Schedule: [system event=MetricsBackupFailure] **[Executes on all nodes in the cluster]** **kpimetricsTrig_52_MetricsBackupFailure →** pMetricsSystemEvent	DEFAULT=ON This is activated when the system event=MetricsBackupFailure occurs. This procedure sends an email to the commonValues.sendTo list and the subscription list. If there is a subscription list then it also saves the record to the METRICS_EVENT_REG_LOG.
Schedule: [system event=MetricsRestoreFailure] **[Executes on all nodes in the cluster]** **kpimetricsTrig_53_MetricsRestoreFailure →** pMetricsSystemEvent	DEFAULT=ON This is activated when the system event=MetricsRestoreFailure occurs. This procedure sends an email to the commonValues.sendTo list and the subscription list. If there is a subscription list then it also saves the record to the METRICS_EVENT_REG_LOG.

© Copyright 2000-2017 TIBCO Software Inc.

Metadata System Helpers Scripts

Location: /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers

This section lists all triggers and load scripts that have been defined to execute various KPImetrics procedures at regular intervals. The default execution frequencies are listed for each trigger. The load scripts have been created to load and aggregate raw data into processed KPImetrics metrics.

Script Name → Resource name	Description
executeMetricsSqlControl	This procedure compares the nodehost/nodeport passed in with what is present in METRICS_SQL_CONTROL and is used to control the execution of the procedure referenced in the CONTROL_NAME field. These rows were inserted by updateTriggers according to the following rules. Both Cache_ALL_RESOURCES and Cache_ALL_USERS are required to be executed on each node of the cluster for the dependent resources to be able to use the data collected. Rules: Cache_ALL_RESOURCES gets executed by dependents: Cache_ALL_RESOURCES Cache_METRICS_SQL_REQUEST_EXEC Cache_METRICS_SQL_REQUEST_EXEC_ADHOC Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS Cache_METRICS_SQL_RESOURCE_LINEAGE pMETRICS_ALL_TABLES_exec Cache_ALL_USERS gets executed by dependents: Cache_ALL_USERS pMETRICS_ALL_TABLES_exec
freeUnusedMemory	This procedure is used to free unused memory. It gets invoked by Cache_METADATA_Tables
getClusterHostnamesDiff	The purpose of this procedure is to compare the current hostname for the current server with names in the cluster to determine the difference of those names. This difference will be used to create and use stage tables for the P_METRICS_ALL_TABLES procedure to use. It requires a unique stage table for each node in the cluster because it performs a table truncate during its processing. hostname1 hostname2 The difference for the current hostname=1 hostname_a hostname_b

© Copyright 2000-2017 TIBCO Software Inc.

	The difference for the current hostname=a
getDatasourceConfiguration → /shared/ASAssets/Utilities/…/ getBasicResourceCursor → getBasicResourceXSLT	This procedure is used to get the current data source configuration for various commonValue settings, paths and derived paths. It invokes a couple of other procedures to get the data source type.
getDatasourceDependencyMetadata	This procedure is used to produce the dependency metadata for a given input resource path. It returns the datasource id, path and dependency resource path along with the lineage for that path. For each new datasource found, it will return a row with the afor-mentioned results. The input resource path may be from /services/databases/ or /shared/ paths. This procedure uses /lib/resource/GetColumnDependencies when a resource path is of type LINK or TABLE. When a resource path is PROCEDURE or a PROCEDURE is found during processing of GetColumnDependencies(), the algorithm will utilize getResourceLineageDirectRecursive() to complete the lineage thus providing a full picture of lineage for resources containing TABLE and PROCEDURE types within the lineage.
getServerNames → getServerNamesXSLT	Get the list of server names for a single node or cluster.
getSysRequestsProcedureName	Get the START_TIME for the procedure name passed in from SYS_REQUESTS. This confirms whether the procedure is actively running in TDV memory space or not.
getVersion	This procedure is used to determine which KPImetrics DDL script version should be used based on the DV version. DV 7.x ==> DDL <script_name>_700 DV 8.x ==> DDL <script_name>_800 The event that would cause a different DDL script to be created would be a change in the base metrics product. For example, if Tibco engineering decided to change the base tables, then a new <script_name>_xxx would be required. The DV version in which the changes are release would dictate the DDL script version and its content.
p15MinutesIncrements	This procedure returns a cursor of 15 minute increments given a starting timestamp that is passed in. It generates one full day of 15 increments for a total of 53 records.
pGetEmailSubscriptions → pGetDomainUsers → [pGetDomainsXSLT,pGetDomainUsersXSLT]	This procedure constructs a list of emails which is used to send error or informational emails based on subscription to DV groups or LDAP groups.
pGetSystemInformation	This script is used to get cluster name and server name from /lib/util/getProperties() built in function. This function is used in load scripts and by custom logger to get server name.
pStartWorkflow	This script is invoked from each of the load scripts and marks the start of a workflow in the METRICS_CIS_WORKFLOW table. For a given workflow, the script takes workflow name as an input and returns the next workflow start and end time.
pEndWorkflow	Like pStartWorkflow script, this script is also called from other load scripts and marks the end of a workflow by updating the METRICS_CIS_WORKFLOW table when a workflow finishes. This script takes workflow name, workflow start and end times, workflow status and number of rows affected as input and updates the METRICS_CIS_WORKFLOW table.
pUpdateEventRegLog	This procedure is used to insert/update rows in the METRICS_EVENT_REG_LOG and METRICS_EVENT_REG_LOG_LINEAGE tables.
verifyProcessRunningMetricsJobDetails	Check for actively running processes for the following tables.

© Copyright 2000-2017 TIBCO Software Inc.

Baseline Table Load Architecture

This section outlines the table load architecture for the three baseline tables described below.

Baseline Table Accessibilty and Parallel Table Load Architecture

This section describes the architecture used to keep the three baseline tables accessible while new data is being loaded. These are the only tables that have this requirement. These tables are used by other load scripts and must be accessible for user queries and joining with data from other load scripts. The three tables are:

- *METRICS_ALL_RESOURCES* – must be accessible all the time and loaded in parallel
- *METRICS_ALL_USERS* – must be accessible all the time and loaded in parallel
- *METRICS_LDAP_PERSON* – must be accessible all the time. Only 1 node in the cluster will load it because the data is common to all nodes

The following architecture is being used for METRICS_ALL_RESOURCES and METRICS_ALL_USERS. This example shows a 2-node cluster with no timekeeper. For each additional node, there would be an additional _N## view and 2 tables N##_T1 and N##_T2.

- TDV references the same view name in the database.
- The top-level database view is a UNION ALL of all the nodes.
- Each node-level view designated with _N## points to the current _N##_T1 or _N##_T2 that is currently loaded with data. The other table has been truncated and contains no data.
- When Cache_ALL_RESOURCES or Cache_ALL_USERS executes, it determines which table [T1 or T2] has no data. It designates the table with no data as the load table. It designates the table with data as the truncate table. The following shows an example and the steps that are performed.

  Supposition: T1 contains rows. T2 is empty.

  a. T1 remains available for queries since it has data.
  b. Each node in the cluster wakes up at the same time on a trigger timer and executes the same code.
  c. Insert data:
     i. NODE 01: Perform INSERT INTO METRICS_ALL_RESOURCES_N01_T2 SELECT * FROM /shared/ASAssets/KPImetrics/Physical/Metadata/System/ALL_RESOURCES
     ii. NODE 02: Perform INSERT INTO METRICS_ALL_RESOURCES_N02_T2 SELECT * FROM /shared/ASAssets/KPImetrics/Physical/Metadata/System/ALL_RESOURCES
  j. Execute table statistics
     i. NODE 01: UPDATE STATISTICS "schema"."METRICS_ALL_RESOURCES_N01_T2"
     ii. NODE 02: UPDATE STATISTICS "schema"."METRICS_ALL_RESOURCES_N02_T2"
  k. Alter View to repoint to newly loaded table
     i. NODE 01: ALTER VIEW "schema"."METRICS_ALL_RESOURCES_N01" AS SELECT * FROM "schema"."METRICS_ALL_RESOURCES_N01_T2"
     ii. NODE 02: ALTER VIEW "schema"."METRICS_ALL_RESOURCES_N02" AS SELECT * FROM "schema"."METRICS_ALL_RESOURCES_N02_T2"
  l. Truncate old table

© Copyright 2000-2017 TIBCO Software Inc.

i. NODE 01: TRUNCATE TABLE "schema"."METRICS_ALL_RESOURCES_N01_T1"
ii. NODE 02: TRUNCATE TABLE "schema"."METRICS_ALL_RESOURCES_N02_T1"

In the diagram below, there are 2 physical tables created [T1 and T2] for each node in the cluster [N01, N02, etc]. There is a database view for each node [<table>_N01] that is mapped to the table that contains the most current data. The loading simply alternates between the two tables. There is another view that is a UNION ALL view that unions each of the node views. The name of this view is the same as the base view in TDV. In the example below the TDV view METRICS_ALL_RESOURCES access the database view METRICS_ALL_RESOURCES which is the UNION ALL of all the database node views. This abstraction allows TDV to load the data into the empty table and switch the node view to the newly loaded table and truncate the old one. This allows each node to execute in parallel and load their own tables and switch their own view when they have completed. It also maintains data accessibility during this load period.

```
TDV        : VIEW                          [METRICS_ALL_RESOURCES]
DB         : VIEW (UNION ALL) |___________[METRICS_ALL_RESOURCES]________|
                              |                                           |
DB         : VIEW             |___[METRICS_ALL_RESOURCES_N01]            |___[METRICS_ALL_RESOURCES_N02]
switch between T1 and T2 -->  |                                           |
DB         : TABLE 1          |---[METRICS_ALL_RESOURCES_N01_T1]         |---[METRICS_ALL_RESOURCES_N02_T1]
DB         : TABLE 2          |---[METRICS_ALL_RESOURCES_N01_T2]         |---[METRICS_ALL_RESOURCES_N02_T2]
```

For METRICS_LDAP_PERSON, the architecture is slightly different because only 1 node in a cluster loads the LDAP data for all other nodes. The data is not TDV node specific.

In the diagram below, there are 2 physical tables created [T1 and T2]. There is a database view that is mapped to the table that contains the most current data. The loading simply alternates between the two tables. TDV accesses the database view.

```
TDV        : VIEW                          [METRICS_LDAP_PERSON]
DB         : VIEW             |________[METRICS_LDAP_PERSON]
switch between T1 and T2 -->|
DB         : TABLE 1          |---[METRICS_LDAP_PERSON_T1]
DB         : TABLE 2          |---[METRICS_LDAP_PERSON_T2]
```

© Copyright 2000-2017 TIBCO Software Inc.

Data Transfer Scripts

This section outlines the Oracle and SQL Server data transfer scripts.

Physical Oracle Data Transfer Script

Location: /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/ 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script

This section describes the actions and SQL taken in the P_METRICS_ALL_TABLES data transfer script. The purpose of this script is to transfer data from the collection tables to the history tables using native Oracle SQL script. Stage tables are used to prepare and augment the collection data prior to inserting into the history tables. The result of this is that only “inserts” are performed on the history tables. All of the insert/update/deletes that are required are performed on the stage tables.

metrics_sessions

1. **Get the min/max sessionid and logintime**

```
select min("sessionid") minsessionid, min("logintime") minlogintime, max("sessionid") maxsessionid,
max("logintime") maxlogintime from '"||dataSourceSchemaName||'"."||metrics_sessions_collection||'" where
"nodehost" = c_node_host and "nodeport" = c_node_port;
```

2. **Insert into the metrics_sessions_stg stage table from the metrics_sessions collection table**

```
insert into '"||dataSourceSchemaName||'"."||metrics_sessions_stg||'"
  select ms.*,
      LU."USER_ID", LU."user", LU."domain",
-- Perform LDAP userkey join transformation.  The transformation value in userKeyTransformation is
   provided at the time this procedure is created which is during installation.
      '||REPLACE(userKeyTransform, '"user"', 'LU."user"')||' as "userkey",
      '||REPLACE(domainKeyTransform, '"domain"', 'LU."domain"')||' as "domainkey",
      null "fullName", null "lastName" , null "firstName", null "mail", null "employeeNumber", null
 "employeeID", null "telephoneNumber", null "friendlyCountryName"
  from '"||dataSourceSchemaName||'"."||metrics_sessions_collection||'" ms
 -- Join with metrics_resources_usage to get the user
LEFT OUTER JOIN (
   select distinct mruc."nodehost", mruc."nodeport", mruc."sessionid", mruc."user", mruc."domain",
 mau."USER_ID"
   FROM '"||dataSourceSchemaName||'"."||metrics_resources_usage_collection||'" mruc
 -- Join with METRICS_ALL_USERS to get the latest (max LOAD_TIME) userid
   LEFT OUTER JOIN '"||dataSourceSchemaName||'"."METRICS_ALL_USERS" mau
     ON mau."USERNAME" = mruc."user"
     AND mau."DOMAIN_NAME" = mruc."domain"
     AND mau."NODE_HOST" = mruc."nodehost"
     AND mau."NODE_PORT" = mruc."nodeport"
  ) LU
  ON LU."sessionid" = ms."sessionid"
  AND LU."nodehost" = ms."nodehost"
  AND LU."nodeport" = ms."nodeport"
  where ms."logintime" <= maxlogintime
   and ms."sessionid" <= maxsessionid
   and ms."nodehost" = in_node_host
   and ms."nodeport" = in_node_port
   and NOT EXISTS (
    select *
    from '"||dataSourceSchemaName||'"."||metrics_sessions_hist||'" ms1
    where ms1."nodehost"  = ms."nodehost"
```

© Copyright 2000-2017 TIBCO Software Inc.

```
  and ms1."nodeport"  = ms."nodeport"
  and ms1."sessionid" = ms."sessionid"
  and ms1."logintime" = ms."logintime"
  and ms1."type" = ms."type"
);
```

3. **Update the metrics_sessions_stg stage table with user information from METRICS_LDAP_PERSON using the "userkey" to join.**

```
UPDATE "'||dataSourceSchemaName||'"."'||metrics_sessions_stg||'" msh1
    SET ("fullname", "lastname", "firstname", "mail", "employeenumber", "employeeid", "telephonenumber",
"friendlycountryname") =
      (SELECT DISTINCT mlp."fullName", mlp."lastName", mlp."firstName", mlp."mail",
mlp."employeeNumber", mlp."employeeID", mlp."telephoneNumber", mlp."friendlyCountryName"
        FROM "'||dataSourceSchemaName||'"."'||metrics_sessions_stg||'" msh2
        LEFT OUTER JOIN "'||dataSourceSchemaName||'"."METRICS_LDAP_PERSON" mlp
          ON mlp."userkey" = msh2."userkey"
          AND mlp."domainkey" = msh2."domainkey"
        WHERE msh1."nodehost"  = msh2."nodehost"
          AND msh1."nodeport"  = msh2."nodeport"
          AND msh1."sessionid" = msh2."sessionid"
          AND msh1."logintime" = msh2."logintime"
          AND msh1."type" = msh2."type"
      )
      WHERE msh1."logintime" >= minlogintime
        AND msh1."sessionid" >= minsessionid
        AND msh1."nodehost" = in_node_host
        AND msh1."nodeport" = in_node_port
        AND msh1."user" is not null
        AND msh1."userkey" is not null
        AND msh1."mail" is null
        AND msh1."lastname" is null;
```

4. **Insert into metrics_sessions_hist history from metrics_sessions_stg stage**

```
INSERT INTO "'||dataSourceSchemaName||'"."'||metrics_sessions_hist||'"
    SELECT * FROM "'||dataSourceSchemaName||'"."'||metrics_sessions_stg||'"
    where "nodehost" = in_node_host
      and "nodeport" = in_node_port;
```

5. **[DEPRECATED] Delete this batch of data from the merics_sessions collection table that was just processed being careful not to delete any new rows.**

```
delete from "'||dataSourceSchemaName||'"."'||metrics_sessions_collection||'"
    where "logintime" <= maxlogintime
      and "sessionid" <= maxsessionid
      and "nodehost" = in_node_host
      and "nodeport" = in_node_port;
```

metrics_resources_usage

1. **Get the min/max requestid and starttime**

```
select min("requestid") minrequestid, min("starttime") minstarttime, max("requestid") maxrequestid,
   max("starttime") maxstarttime
      from "'||dataSourceSchemaName||'"."'||metrics_resources_usage_collection||'"
      where "nodehost" = c_node_host and "nodeport" = c_node_port;
```

© Copyright 2000-2017 TIBCO Software Inc.

2. **Insert the new rows into the metrics_resources_usage_stg stage from the metrics_resources_usage collection where not exists in metrics_resources_usage_hist history and not in METRICS_JOB_FILTERS.**

```
INSERT INTO '"||dataSourceSchemaName||'"."||metrics_resources_usage_stg||'"
  SELECT
      -- Insert collection columns
        mruc."cluster"
      , mruc."nodehost"
      , mruc."nodeport"
      , mruc."sessionid"
      , mruc."user"
      , mruc."domain"
      , mruc."group"
      , mruc."requestid"
      , mruc."parentid"
      , mruc."datasourcepath"
      , mruc."datasourcetype"
      , mruc."resourcepath"
      , mruc."resourcetype"
      , mruc."resourceguid"
      , mruc."resourcekind"
      , mruc."starttime"
      , mruc."endtime"
      -- Insert expanded information
      , mar."RESOURCE_ORIGIN" "resourceorigin"
      , mar."RESOURCE_ID" "resourceid"
      , mar."DATASERVICE_NAME" "dataservicename"
      , mar."RESOURCE_NAME" "resourcename"
      , mar."SCHEMA_NAME" "parentname"
      , mar."CATALOG_NAME" "grandparentname"
      , mar."CATEGORY_NAME" "categoryname"
      , mar."PARENT_PATH" "parentpath"
      -- Insert expanded user information
      , mau."USER_ID" "userid"
      , '||REPLACE(userKeyTransform, '"user"', 'mruc."user"')||' as "userkey"
      , '||REPLACE(domainKeyTransform, '"domain"', mruc."domain")||' as "domainkey"
      , null "fullName", null "lastName" , null "firstName", null "mail", null "employeeNumber", null "employeeID", null
"telephoneNumber", null "friendlyCountryName"
  FROM '"||dataSourceSchemaName||'"."||metrics_resources_usage_collection||'" mruc
    -- METRICS_ALL_USERS
    LEFT OUTER JOIN '"||dataSourceSchemaName||'"."METRICS_ALL_USERS" mau
      ON  mau."USERNAME"    = mruc."user"
      AND mau."DOMAIN_NAME" = mruc."domain"
      AND mau."NODE_HOST"   = mruc."nodehost"
      AND mau."NODE_PORT"   = mruc."nodeport"
    -- METRICS_ALL_RESOURCES
    LEFT OUTER JOIN '"||dataSourceSchemaName||'"."METRICS_ALL_RESOURCES" mar
      ON  mar."RESOURCEPATH"  = mruc."resourcepath"
      AND mar."RESOURCE_TYPE" = mruc."resourcetype"
      AND mar."NODE_HOST"     = mruc."nodehost"
      AND mar."NODE_PORT"     = mruc."nodeport"
    WHERE mruc."nodehost"  = in_node_host
      AND mruc."nodeport"  = in_node_port
      AND mruc."starttime" <= maxstarttime_mru
      AND mruc."requestid" <= maxrequestid_mr
      -- Do not insert rows matching the filter rows in METRICS_JOB_FILTERS
      AND (mruc."user", mruc."domain", mruc."resourcekind") NOT IN (select "USER", "DOMAIN", "RESOURCE_KIND"
FROM '"||dataSourceSchemaName||'"."METRICS_JOB_FILTERS" WHERE "ENV_TYPE" = in_env_type)
      -- Do not insert rows if they exist in metrics_resources_usage_hist
      AND NOT EXISTS (
        SELECT 1
        FROM '"||dataSourceSchemaName||'"."||metrics_resources_usage_hist||'" mruh
        WHERE mruc."nodehost"  = mruh."nodehost"
          AND mruc."nodeport"  = mruh."nodeport"
          AND mruc."starttime" = mruh."starttime"
          AND mruc."requestid" = mruh."requestid"
```

© Copyright 2000-2017 TIBCO Software Inc.

```
);
```

6. **Update the metrics_resources_usage_stg stage table with user information from METRICS_LDAP_PERSON using the "userkey" to join.**

```
UPDATE "'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'" mruh1
    SET ("fullname", "lastname", "firstname", "mail", "employeenumber", "employeeid", "telephonenumber",
"friendlycountryname") =
        (SELECT DISTINCT mlp."fullName", mlp."lastName", mlp."firstName", mlp."mail",
mlp."employeeNumber", mlp."employeeID", mlp."telephoneNumber", mlp."friendlyCountryName"
          FROM "'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'" mruh2
          LEFT OUTER JOIN "'||dataSourceSchemaName||'"."METRICS_LDAP_PERSON" mlp
            ON mlp."userkey" = mruh2 ."userkey"
            AND mlp."domainkey" = mruh2."domainkey"
          WHERE mruh1 ."nodehost"  = mruh2 ."nodehost"
            AND mruh1 ."nodeport"  = mruh2 ."nodeport"
            AND mruh1 ."requestid" = mruh2 ."requestid"
            AND mruh1 ."starttime" = mruh2 ."starttime"
        )
        WHERE mruh1 ."starttime" >= minstarttime_mru
          AND mruh1 ."requestid" >= minrequestid_mru
          AND mruh1 ."nodehost" = in_node_host
          AND mruh1 ."nodeport" = in_node_port
          AND mruh1 ."user" is not null
          AND mruh1 ."userkey" is not null
          AND mruh1 ."mail" is null
          AND mruh1 ."lastname" is null;
```

7. **Insert into metrics_resources_usage_hist history from metrics_resources_usage_stg stage**

```
INSERT INTO "'||dataSourceSchemaName||'"."'||metrics_resources_usage_hist||'"
    SELECT * FROM "'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'"
    where "nodehost" = in_node_host
      and "nodeport" = in_node_port;
```

8. **[DEPRECATED] Delete the metrics_resources_usage collection remaining data**
   a. **Note: if the kpimetricsTrig_33_DeleteCollection trigger is still running then this section is skipped.**

```
delete from "'||dataSourceSchemaName||'"."'||metrics_resources_usage_collection||'"
    where "nodehost" = in_node_host
      and "nodeport" = in_node_port
      and "starttime" <= maxstarttime_mru
      and "requestid" <= maxrequestid_mru;
```

metrics_requests

1. **Get the min/max requestid and starttime**

```
select min("requestid") minrequestid, min("starttime") minstarttime, max("requestid") maxrequestid,
   max("starttime") maxstarttime
      from "'||dataSourceSchemaName||'"."'||metrics_requests_collection||'"
      where "nodehost" = c_node_host and "nodeport" = c_node_port;
```

2. **Insert the new rows into the metrics_requests_stg stage from the metrics_requests collection where not exists in metrics_requests_hist history and exists in metrics_resources_usage_stg.**

© Copyright 2000-2017 TIBCO Software Inc.

```
-- A cursor is created in order to be able to loop through the records and commit a batch at a time.  This was
done so as to minimize the UNDO tablespace and archiver rollback logs.  When Oracle archiver reaches a
configured length of time and the query has not finished it tries to rollback.  If there is not enough UNDO
tablespace, the query fails.   For this particular query it may be necessary to greatly increase the UNDO
tablespace and increase the time for the archiver to 2 hours.

cursor c_metrics_requests(c_node_host varchar2, c_node_port number, c_maxrequestid number,
c_maxstarttime timestamp) is
SELECT mrc.*,
          null "dataservicename", null "resourcekind",
          LU."USER_ID", LU."user", LU."domain",
          '||REPLACE(userKeyTransform, '"user"', 'LU."user"')||' as "userkey",
          '||REPLACE(domainKeyTransform, '"domain"', 'LU."domain"')||' as "domainkey",
          null "fullName", null "lastName" , null "firstName", null "mail", null "employeeNumber", null
"employeeID", null "telephoneNumber", null "friendlyCountryName"
FROM "'||dataSourceSchemaName||'"."'||metrics_requests_collection||'" mrc
-- METRICS_ALL_USERS
LEFT OUTER JOIN
        (SELECT DISTINCT mruc."nodehost", mruc."nodeport", mruc."requestid", mruc."user",
mruc."domain", mau."USER_ID"
        FROM "'||dataSourceSchemaName||'"."'||metrics_resources_usage_collection||'" mruc
        LEFT OUTER JOIN "'||dataSourceSchemaName||'"."METRICS_ALL_USERS" mau
                    ON  mau."USERNAME"    = mruc."user"
                    AND mau."DOMAIN_NAME" = mruc."domain"
                    AND mau."NODE_HOST"   = mruc."nodehost"
                    AND mau."NODE_PORT"   = mruc."nodeport"
          ) LU
          ON  LU."requestid" = mrc."requestid"
          AND LU."nodehost"  = mrc."nodehost"
          AND LU."nodeport"  = mrc."nodeport"
        WHERE mrc."nodehost" = c_node_host
          AND mrc."nodeport" = c_node_port
          AND mrc."requestid" <= c_maxrequestid
          AND mrc."starttime" <= c_maxstarttime
          -- Only insert metrics_requests collection rows with a corresponding row in the
metrics_resources_usage_stg
          --    No point in capturing these rows without relevant resource usage information.
          AND EXISTS (
                              SELECT 1 FROM
"'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'" mru
                              WHERE mru."nodehost"  = mrc."nodehost"
                                AND mru."nodeport"  = mrc."nodeport"
                                AND mru."requestid" = mrc."requestid"
                                AND mru."starttime" = mrc."starttime"
          );

TYPE c_metrics_requests_T is table of c_metrics_requests%rowtype;
c_metrics_requests_array c_metrics_requests_T;

-- Perform the insert in batches that get committed every "batchInsertMax" to free up UNDO space.
--    The fetch size and commit size are the same to keep the logic clean.
batchcount := 0;
begintimestamp3 := SYSTIMESTAMP;
open c_metrics_requests(in_node_host, in_node_port, maxrequestid_mr, maxstarttime_mr);

loop fetch c_metrics_requests bulk collect into c_metrics_requests_array limit batchInsertMax;
   -- Calculate fetch collection duration
   endtimestamp := SYSTIMESTAMP;
   fetchDuration_mr := endtimestamp - begintimestamp3;
   durationStr := replace(replace(cast(fetchDuration_mr as varchar2), "000000000 ", "0 "), "+", "");
```

© Copyright 2000-2017 TIBCO Software Inc.

```
    fetchDurationStr_mr := substr(durationStr, 1, instr(durationStr, ".") + 3);

    -- Insert the rows a batch at a time
    begintimestamp3 := SYSTIMESTAMP;
    forall i in 1 .. c_metrics_requests_array.count
        INSERT INTO "'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" VALUES
c_metrics_requests_array(i);

    uncommitted := uncommitted + sql%rowcount;
    insertrows_mr := insertrows_mr + sql%rowcount;
    batchcount := batchcount + 1;
    -- Calculate insert stage duration
    endtimestamp := SYSTIMESTAMP;
    insertStgDuration_mr := endtimestamp - begintimestamp3;
    durationStr := replace(replace(cast(insertStgDuration_mr as varchar2), "000000000 ", "0 "), "+", "");
    insertStgDurationStr_mr := substr(durationStr, 1, instr(durationStr, ".") + 3);
    -- Update a status row into the METRICS_JOB_DETAILS table for number of rows affected
    statusmsg := "INSERT STAGE: '||metrics_requests_stg||' B#="||batchcount||" T#="||insertrows_mr||"
FETCH="||fetchDurationStr_mr||" INSERT="||insertStgDurationStr_mr;

   update "'||dataSourceSchemaName||'"."METRICS_JOB_DETAILS"
                set CURRENT_OPERATION = statusmsg
                 where REQ_START_DATE = begintimestamp
                and JOB_TABLE_NAME = '"||metrics_requests_collection||'"
                and NODE_HOST = in_node_host
                and NODE_PORT = in_node_port;
    exit when c_metrics_requests_array.count < batchInsertMax;
    if (uncommitted >= batchInsertMax) then
       commit;
       uncommitted := 0;
    end if;
    begintimestamp3 := SYSTIMESTAMP;
end loop;
if (uncommitted > 0) then
    commit;
end if;
close c_metrics_requests;

-- Delete any rows from stage that exist in history.  This logic was broken out as a separate piece instead of
 embedding in the cursor query above because in most cases there should be no rows.  This is here as
 more of an insurance policy.

DELETE FROM "'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrc
  -- Remove any rows that already exist in metrics_requests_hist
WHERE mrc."nodehost" = in_node_host
  AND mrc."nodeport" = in_node_port
  AND EXISTS (
     SELECT 1
     FROM "'||dataSourceSchemaName||'"."'||metrics_requests_hist||'" mrh
     WHERE mrc."nodehost"  = mrh."nodehost"
       AND mrc."nodeport"  = mrh."nodeport"
       AND mrc."starttime" = mrh."starttime"
       AND mrc."requestid" = mrh."requestid"
     );
deleterows_mr := sql%rowcount;
commit;
```

3. **Update the new rows from the metrics_requests_stg stage with METRICS_LDAP_PERSON using the "userkey" to join.**

© Copyright 2000-2017 TIBCO Software Inc.

```
UPDATE "'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrh1
    SET ("fullname", "lastname", "firstname", "mail", "employeenumber", "employeeid", "telephonenumber",
  "friendlycountryname") =
      (SELECT DISTINCT mlp."fullName", mlp."lastName", mlp."firstName", mlp."mail",
  mlp."employeeNumber", mlp."employeeID", mlp."telephoneNumber", mlp."friendlyCountryName"
        FROM "'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrh2
        LEFT OUTER JOIN "'||dataSourceSchemaName||'"."METRICS_LDAP_PERSON" mlp
          ON mlp."userkey" = mrh2."userkey"
          AND mlp.”domainkey” = mrh2.”domainkey”
        WHERE mrh1."nodehost"  = mrh2."nodehost"
         AND mrh1."nodeport"  = mrh2."nodeport"
         AND mrh1."requestid" = mrh2."requestid"
         AND mrh1."starttime" = mrh2."starttime"
      )
      WHERE mrh1."starttime" >= minstarttime_mr
        AND mrh1."requestid" >= minrequestid_mr
        AND mrh1."nodehost" = in_node_host
        AND mrh1."nodeport" = in_node_port
        AND mrh1."user" is not null
        AND mrh1."userkey" is not null
        AND mrh1."mail" is null
        AND mrh1."lastname" is null;
```

4. **Update metrics_requests_stg stage [resourcekind and dataservicename] joined with metrics_resources_usage_stg.**
   a. Use the resourcekind and dataservicename from metrics_resources_usage_stg.
   b. This process is to insure that user defined requests are updated first in the event that metrics_resoruces_usage contains multiple rows for the same requestid where the resourcekind spans both [user defined] and [system].
   c. If multiple distinct "requestid, resourcekind and dataservicename" rows are found the algorithm orders by mruh.requestid, mruh.resourcekind desc, mruh.dataservicename so that "user defined" is sorted before "system" as "user defined" has higher priority.
   d. Only one record can be chosen.

   <u>Loop through the records using the following SQL</u>

```
select distinct mruh."requestid", mrh."starttime", mrh."endtime", mrh."status", mruh."resourcekind",
mruh."dataservicename"
from "'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrh
   inner join "'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'" mruh
                on mrh."requestid" = mruh."requestid"
               and mrh."starttime" = mruh."starttime"
               and mrh."nodehost" = mruh."nodehost"
               and mrh."nodeport" = mruh."nodeport"
                          where mruh."resourcekind" is not null
                            and mruh."dataservicename" is not null
            and mrh."nodehost" = c_node_host
            and mrh."nodeport" = c_node_port
                            and (mrh."resourcekind" is null or mrh."dataservicename" is null)
            and mrh."requestid" >= c_minrequestid and mrh."requestid" <= c_maxrequestid
            and mrh."starttime" >= c_minstarttime and mrh."starttime" <= c_maxstarttime
                order by mruh."requestid", mruh."resourcekind" desc, mruh."dataservicename";
```

   <u>Insert records into metrics_requests_stg table in pre-defined batches</u>

```
insert into "'||dataSourceSchemaName||'"."'||metrics_requests_stg_upd||'" values(t."requestid",
t."starttime", t."endtime", t."status", t."resourcekind", t."dataservicename", in_node_host, in_node_port);
```

© Copyright 2000-2017 TIBCO Software Inc.

Update the metrics_requests_stg with the single row for each requestid using the batch in the metrics_request_stg_upd stage table. The min and max requestid and starttime are important to establish boundaries for the update. Without them the resourcekind and dataservicename for rows not in the stage update table are set to null.

```
update '"||dataSourceSchemaName||'"."||metrics_requests_stg||'" mrh
      set (mrh."resourcekind", mrh."dataservicename") =
          (select mrht."resourcekind", mrht."dataservicename"
                from '"||dataSourceSchemaName||'"."||metrics_requests_stg_upd||'" mrht
                where mrh."requestid" = mrht."requestid"
                  and mrh."starttime" = mrht."starttime"
                  and mrh."endtime" = mrht."endtime"
                  and mrh."status" = mrht."status"
                  and mrh."nodehost" = mrht."nodehost"
                  and mrh."nodeport" = mrht."nodeport"
                  and mrht."nodehost" = in_node_host
                  and mrht."nodeport" = in_node_port
          )
where mrh."resourcekind" is null
      and mrh."dataservicename" is null
      and mrh."nodehost" = in_node_host
      and mrh."nodeport" = in_node_port
      and mrh."requestid" >= minrequestid_mrb
      and mrh."requestid" <= maxrequestid_mrb
      and mrh."starttime" >= minstarttime_mrb
      and mrh."starttime" <= maxstarttime_mrb;
```

Delete only rows associated with the nodehost and nodeport that is executing this script

```
delete from '"||dataSourceSchemaName||'"."||metrics_requests_stg_upd||'" where "nodehost" =
in_node_host and "nodeport" = in_node_port;
```

End of Loop

5. **Insert into the metrics_requests_hist history from the metrics_requests_stg stage**

```
INSERT INTO '"||dataSourceSchemaName||'"."||metrics_requests_hist||'"
    SELECT * FROM '"||dataSourceSchemaName||'"."||metrics_requests_stg||'"
    where "nodehost" = in_node_host
      and "nodeport" = in_node_port;
```

6. **[DEPRECATED] Delete the remainder of metrics_requests collection rows**
   a. **Note: if the kpimetricsTrig_33_DeleteCollection trigger is still running then this section is skipped.**

```
delete from '"||dataSourceSchemaName||'"."||metrics_requests_collection||'"
    where "nodehost" = in_node_host
     and "nodeport" = in_node_port
     and "starttime" <= maxstarttime_mr
     and "requestid" <= maxrequestid_mr;
```

© Copyright 2000-2017 TIBCO Software Inc.

Physical SQL Server Data Transfer Script

Location: /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/ 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script

This section describes the actions and SQL taken in the P_METRICS_ALL_TABLES data transfer script. The purpose of this script is to transfer data from the collection tables to the history tables using native SQL Server SQL script. Stage tables are used to prepare and augment the collection data prior to inserting into the history tables. The result of this is that only “inserts” are performed on the history tables. All of the insert/update/deletes that are required are performed on the stage tables.

metrics_sessions

1. Get the min/max sessionid and logintime

```
select min("sessionid") minsessionid, min("logintime") minlogintime, max("sessionid") maxsessionid,
max("logintime") maxlogintime
      from '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_sessions_collection||'"
      where "nodehost" = @in_node_host and "nodeport" = @in_node_port;
```

2. Insert from metrics_sessions collection table into metrics_sessions_stg stage table

```
insert into '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_sessions_stg||'"
  select ms.*,
      LU."USER_ID", LU."user", LU."domain",
-- Perform LDAP userkey join transformation.  The transformation value in userKeyTransformation is provided
   at the time this procedure is created which is during installation.
      '||REPLACE(userKeyTransform, '"user"', 'LU."user"')||' as "userkey",
      '||REPLACE(domainKeyTransform, '"domain"', 'LU."domain"')||' as "domainkey",
      null "fullName", null "lastName" , null "firstName", null "mail", null "employeeNumber", null "employeeID",
 null "telephoneNumber", null "friendlyCountryName"
  from '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_sessions_collection||'" ms
-- Join with metrics_resources_usage to get the user
  LEFT OUTER JOIN (
    select distinct mruc."nodehost", mruc."nodeport", mruc."sessionid", mruc."user", mruc."domain",
 mau."USER_ID"
    FROM
'"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_resources_usage_collection||'" mruc
 -- Join with METRICS_ALL_USERS to get the latest (max LOAD_TIME) userid
    LEFT OUTER JOIN
'"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."METRICS_ALL_USERS" mau
      ON mau."USERNAME" = mruc."user"
      AND mau."DOMAIN_NAME" = mruc."domain"
      AND mau."NODE_HOST" = mruc."nodehost"
      AND mau."NODE_PORT" = mruc."nodeport"
  ) LU
  ON LU."sessionid" = ms."sessionid"
  AND LU."nodehost" = ms."nodehost"
  AND LU."nodeport" = ms."nodeport"
  where ms."logintime" <= @maxlogintime
    and ms."sessionid" <= @maxsessionid
    and ms."nodehost" = @in_node_host
    and ms."nodeport" = @in_node_port
    and NOT EXISTS (
     select *
     from '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_sessions_hist||'" ms1
```

© Copyright 2000-2017 TIBCO Software Inc.

```
where ms1."nodehost"  = ms."nodehost"
  and ms1."nodeport"  = ms."nodeport"
  and ms1."sessionid" = ms."sessionid"
  and ms1."logintime" = ms."logintime"
  and ms1."type" = ms."type"
);
```

3. **Update the metrics_sessions_stg table with user information from METRICS_LDAP_PERSON using the "userkey" to join.**

```
UPDATE "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_sessions_stg||'"
      SET "fullname" = mlp."fullName",
         "lastname" = mlp."lastName",
         "firstname" = mlp."firstName",
         "mail" = mlp."mail",
         "employeenumber" = mlp."employeeNumber",
         "employeeid" = mlp."employeeID",
         "telephonenumber" = mlp."telephoneNumber",
         "friendlycountryname" = mlp."friendlyCountryName"
       FROM "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_sessions_stg||'" msh
       LEFT OUTER JOIN
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."METRICS_LDAP_PERSON" mlp
          ON mlp."userkey" = msh."userkey"
          AND mlp."domainkey" = msh."domainkey"
       WHERE msh."logintime" >= @minlogintime
         AND msh."sessionid" >= @minsessionid
         AND msh."nodehost" = @in_node_host
         AND msh."nodeport" = @in_node_port
         AND msh."user" is not null
         AND msh."userkey" is not null
         AND msh."mail" is null
         AND msh."lastname" is null;
```

4. **Insert into metrics_sessions_hist history from metrics_sessions_stg stage**

```
INSERT INTO "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_sessions_hist||'"
    SELECT * FROM "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_sessions_stg||'"
    where "nodehost" = in_node_host
      and "nodeport" = in_node_port;
```

5. **[DEPRECATED] Delete this batch of data from the merics_sessions collection table that was just processed being careful not to delete any new rows.**

```
delete from "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_sessions_collection||'"
    where "logintime" <= @maxlogintime
      and "sessionid" <= @maxsessionid
      and "nodehost" = @in_node_host
      and "nodeport" = @in_node_port;
```

metrics_resources_usage

1. **Get the min/max requestid and starttime**

```
select min("requestid") minrequestid, min("starttime") minstarttime, max("requestid") maxrequestid,
   max("starttime") maxstarttime
      from
   "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_collection||'"
      where "nodehost" = @in_node_host and "nodeport" = @in_node_port;
```

© Copyright 2000-2017 TIBCO Software Inc.

2. **Insert the new rows into the metrics_resources_usage_stg stage from the metrics_resources_usage collection where not exists in metrics_resources_usage_hist history and not in METRICS_JOB_FILTERS.**

```
INSERT INTO '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_resources_usage_stg||'"
SELECT
    -- Insert collection columns
      mruc."cluster"
    , mruc."nodehost"
    , mruc."nodeport"
    , mruc."sessionid"
    , mruc."user"
    , mruc."domain"
    , mruc."group"
    , mruc."requestid"
    , mruc."parentid"
    , mruc."datasourcepath"
    , mruc."datasourcetype"
    , mruc."resourcepath"
    , mruc."resourcetype"
    , mruc."resourceguid"
    , mruc."resourcekind"
    , mruc."starttime"
    , mruc."endtime"
    -- Insert expanded information
                                , mar."RESOURCE_ORIGIN" "resourceorigin"
    , mar."RESOURCE_ID" "resourceid"
                                , mar."DATASERVICE_NAME" "dataservicename"
                                , mar."RESOURCE_NAME" "resourcename"
                                , mar."SCHEMA_NAME" "parentname"
                                , mar."CATALOG_NAME" "grandparentname"
    , mar."CATEGORY_NAME" "categoryname"
                                , mar."PARENT_PATH" "parentpath"
    -- Insert expanded user information
    , mau."USER_ID" "userid"
    , '||REPLACE(userKeyTransform, '"user"', 'mruc."user"')||' as "userkey"
    , '||REPLACE(domainKeyTransform, '"domain"', 'mruc."domain"')||' as "domainkey"
    , null "fullName", null "lastName" , null "firstName", null "mail", null "employeeNumber", null "employeeID", null
"telephoneNumber", null "friendlyCountryName"
FROM '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_resources_usage_collection||'" mruc
  -- METRICS_ALL_USERS
  LEFT OUTER JOIN '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."METRICS_ALL_USERS" mau
        ON  mau."USERNAME"    = mruc."user"
        AND mau."DOMAIN_NAME" = mruc."domain"
      AND mau."NODE_HOST"   = mruc."nodehost"
      AND mau."NODE_PORT"   = mruc."nodeport"
  -- METRICS_ALL_RESOURCES
         LEFT OUTER JOIN
'"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."METRICS_ALL_RESOURCES" mar
      ON  mar."RESOURCEPATH"  = mruc."resourcepath"
      AND mar."RESOURCE_TYPE" = mruc."resourcetype"
      AND mar."NODE_HOST"     = mruc."nodehost"
      AND mar."NODE_PORT"     = mruc."nodeport"
WHERE mruc."nodehost" = @in_node_host
  AND mruc."nodeport" = @in_node_port
  AND mruc."starttime" <= @maxstarttime_mru
  AND mruc."requestid" <= @maxrequestid_mr
  -- Do not insert rows matching the filter rows in METRICS_JOB_FILTERS
  AND mruc."requestid" NOT IN
        (SELECT mru."requestid"
          FROM '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_resources_usage_collection||'"
mru
          JOIN '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."METRICS_JOB_FILTERS" mjf
            ON mru."user" = mjf."USER"
            AND mru."domain" = mjf."DOMAIN"
            AND mru."resourcekind" = mjf."RESOURCE_KIND"
                                                AND mru."nodehost" = @in_node_host
```

```
                AND mru."nodeport" = @in_node_port
              WHERE mjf."ENV_TYPE" = @in_env_type )
        -- Do not insert rows if they exist in metrics_resources_usage_hist
        AND NOT EXISTS (
          SELECT 1
          FROM "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_hist||'" mruh
          WHERE mruc."nodehost"  = mruh."nodehost"
            AND mruc."nodeport"  = mruh."nodeport"
            AND mruc."starttime" = mruh."starttime"
            AND mruc."requestid" = mruh."requestid"
        );
```

3. **Update the metrics_resources_usage_stg table with user information from METRICS_LDAP_PERSON using the "userkey" to join.**

```
UPDATE "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'"
        SET "fullname" = mlp."fullName",
            "lastname" = mlp."lastName",
            "firstname" = mlp."firstName",
            "mail" = mlp."mail",
            "employeenumber" = mlp."employeeNumber",
            "employeeid" = mlp."employeeID",
            "telephonenumber" = mlp."telephoneNumber",
            "friendlycountryname" = mlp."friendlyCountryName"
        FROM "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'"
mruh
        LEFT OUTER JOIN
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."METRICS_LDAP_PERSON" mlp
            ON mlp."userkey" = mruh."userkey"
            AND mlp."domainkey" = mruh."domainkey"
        WHERE mruh."starttime" >= @minstarttime_mru
          AND mruh."requestid" >= @minrequestid_mru
          AND mruh."nodehost" = @in_node_host
          AND mruh."nodeport" = @in_node_port
          AND mruh."user" is not null
          AND mruh."userkey" is not null
          AND mruh."mail" is null
          AND mruh."lastname" is null;
```

6. **Insert into metrics_resources_usage_hist history from metrics_resources_usage_stg stage**

```
INSERT INTO
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_hist||'"
    SELECT * FROM
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'"
    where "nodehost" = in_node_host
      and "nodeport" = in_node_port;
```

4. **[DEPRECATED] Delete the metrics_resources_usage collection remaining data**
   a. **Note: if the kpimetricsTrig_33_DeleteCollection trigger is still running then this section is skipped.**

```
delete from
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_collection||'"
      where "nodehost" = @in_node_host
        and "nodeport" = @in_node_port
        and "starttime" <= @maxstarttime_mru
       and "requestid" <= @maxrequestid_mru;
```

metrics_requests

© Copyright 2000-2017 TIBCO Software Inc.

1. **Get the min/max requestid and starttime**

```
select min("requestid") minrequestid, min("starttime") minstarttime, max("requestid") maxrequestid,
   max("starttime") maxstarttime
      from '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_requests_collection||'"
      where "nodehost" = @in_node_host and "nodeport" = @in_node_port;
```

2. **Insert the new rows into the metrics_requests_stg stage from the metrics_requests collection where not exists in metrics_requests_hist history and exists in metrics_resources_usage_stg.**

```
INSERT INTO '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_requests_stg||'"
    SELECT mrc.*,
        null "dataservicename", null "resourcekind",
        LU."USER_ID", LU."user", LU."domain",
        '||REPLACE(userKeyTransform, '"user"', 'LU."user"')||' as "userkey",
        null "fullName", null "lastName" , null "firstName", null "mail", null "employeeNumber", null
  "employeeID", null "telephoneNumber", null "friendlyCountryName"
    FROM '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_requests_collection||'" mrc
    LEFT OUTER JOIN (
      SELECT DISTINCT mruc."nodehost", mruc."nodeport", mruc."requestid", mruc."user", mruc."domain",
  mau."USER_ID"
      FROM
  '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_resources_usage_collection||'"
  mruc
      LEFT OUTER JOIN
  '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."METRICS_ALL_USERS" mau
        ON mau."USERNAME" = mruc."user"
        AND mau."DOMAIN_NAME" = mruc."domain"
        AND mau."NODE_HOST" = mruc."nodehost"
        AND mau."NODE_PORT" = mruc."nodeport"
    ) LU
    ON LU."requestid" = mrc."requestid"
    AND LU."nodehost" = mrc."nodehost"
    AND LU."nodeport" = mrc."nodeport"
  WHERE mrc."nodehost" = @in_node_host
    AND mrc."nodeport" = @in_node_port
    AND mrc."starttime" <= @maxstarttime_mr
    AND mrc."requestid" <= @maxrequestid_mr
    AND NOT EXISTS (
      SELECT 1
      FROM '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_requests_hist||'" mrh
      WHERE mrc."nodehost"  = mrh."nodehost"
        AND mrc."nodeport"  = mrh."nodeport"
        AND mrc."starttime" = mrh."starttime"
        AND mrc."requestid" = mrh."requestid"
      )
    -- Only insert metrics_requests collection rows when a corresponding row exists in the
  metrics_resources_usage_stg
    --    No point in capturing these rows without relevant resource usage information.
    AND EXISTS (
        SELECT 1 FROM
  '"||dataSourceCatalogName||'"."||dataSourceSchemaName||'"."||metrics_resources_usage_stg||'" mru
        WHERE mru."nodehost"  = mrc."nodehost"
          AND mru."nodeport"  = mrc."nodeport"
          AND mru."requestid" = mrc."requestid"
          AND mru."starttime" = mrc."starttime"
    );
```

© Copyright 2000-2017 TIBCO Software Inc.

3. **Update the new rows from the metrics_requests_stg stage with METRICS_LDAP_PERSON data**

```
UPDATE "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'"
      SET "fullname" = mlp."fullName",
         "lastname" = mlp."lastName",
         "firstname" = mlp."firstName",
         "mail" = mlp."mail",
         "employeenumber" = mlp."employeeNumber",
         "employeeid" = mlp."employeeID",
         "telephonenumber" = mlp."telephoneNumber",
         "friendlycountryname" = mlp."friendlyCountryName"
       FROM "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrh
       LEFT OUTER JOIN
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."METRICS_LDAP_PERSON" mlp
            ON mlp."userkey" = mrh."userkey"
             AND mlp."domainkey" = mrh."domainkey"
       WHERE mrh."starttime" >= @minstarttime_mr
         AND mrh."requestid" >= @minrequestid_mr
         AND mrh."nodehost" = @in_node_host
         AND mrh."nodeport" = @in_node_port
         AND mrh."user" is not null
         AND mrh."userkey" is not null
         AND mrh."mail" is null
         AND mrh."lastname" is null;
```

4. **Update metrics_requests_stg stage [resourcekind and dataservicename] joined with metrics_resources_usage_stg.**
   a. Use the resourcekind and dataservicename from metrics_resources_usage_stg.
   b. This process is to ensure that user defined requests are updated first in the event that metrics_resoruces_usage contains multiple rows for the same requestid where the resourcekind spans both [user defined] and [system].
   c. If multiple distinct "requestid, resourcekind and dataservicename" rows are found the algorithm orders by mruh.requestid, mruh.resourcekind desc, mruh.dataservicename so that "user defined" is sorted before "system" as "user defined" has higher priority.
   d. Only one record can be chosen.

<u>Update where resourcekind='user defined'</u>

```
update "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'"
           set "resourcekind" = mruh."resourcekind",
               "dataservicename" = mruh."dataservicename"
         from "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrh
           inner join
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'" mruh
          on mruh."requestid" = mrh."requestid"
          and mruh."starttime" = mrh."starttime"
          and mruh."nodehost" = mrh."nodehost"
          and mruh."nodeport" = mrh."nodeport"
         where mrh."nodehost" = @in_node_host
          and mrh."nodeport" = @in_node_port
          and mrh."starttime" <= @maxstarttime_mr
          and mrh."requestid" <= @maxrequestid_mr
          and mrh."resourcekind" is null
          and mrh."dataservicename" is null
          and mruh."dataservicename" is not null
          and mruh."resourcekind" = "user defined"; -- [user defined, system]
```

© Copyright 2000-2017 TIBCO Software Inc.

Update where resourcekind='system'

```
update "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'"
        set "resourcekind" = mruh."resourcekind",
            "dataservicename" = mruh."dataservicename"
      from "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'" mrh
        inner join
"'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_resources_usage_stg||'" mruh
        on mruh."requestid" = mrh."requestid"
        and mruh."starttime" = mrh."starttime"
        and mruh."nodehost" = mrh."nodehost"
        and mruh."nodeport" = mrh."nodeport"
      where mrh."nodehost" = @in_node_host
        and mrh."nodeport" = @in_node_port
        and mrh."starttime" <= @maxstarttime_mr
        and mrh."requestid" <= @maxrequestid_mr
        and mrh."resourcekind" is null
        and mrh."dataservicename" is null
        and mruh."dataservicename" is not null
        and mruh."resourcekind" = "system"; -- [user defined, system]
```

7. Insert into metrics_requests_hist history from metrics_requests_stg stage

```
INSERT INTO "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_hist||'"
    SELECT * FROM "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_stg||'"
    where "nodehost" = in_node_host
     and "nodeport" = in_node_port;
```

5. [DEPRECATED] Delete the remainder of metrics_requests collection rows

a. Note: if the kpimetricsTrig_33_DeleteCollection trigger is still running then this section is skipped.

```
delete from "'||dataSourceCatalogName||'"."'||dataSourceSchemaName||'"."'||metrics_requests_collection||'"
    where "nodehost" = @in_node_host
     and "nodeport" = @in_node_port
     and "starttime" <= @maxstarttime_mr
     and "requestid" <= @maxrequestid_mr;
```

© Copyright 2000-2017 TIBCO Software Inc.

8 Release Notes

This section provides information on what has changed in this release.

Added or Modified in this Release

This section provides bullet points on what has been added or changed in this release.

Release 2020Q202 [May 1 2020]

- Removed all [Arch] tables as partitioning for each table has been added so that the current and historical data will reside in the same table in an interval of 366 days.
- Only supporting SQL Server 2016 or SQL Server Azure due to native SQL requirements “truncate table <table> with(partitions(n)). The ony datasource that is used is KPI_sqlserver_2016. The other 2 datasources KPI_sqlserver_2012 and KPI_sqlserver_2014 have been dropped. This was necessary to simplify this offering.

Release 2020Q201 [Apr 6 2020]

- Release 2020Q201
  - Patch includes all /System/ClusterSafeCache and /System/Helpers
  - Added check for a dedicated time keeper node so as to not execute procedures on it.
  - Fixed issue with recognizing a cluster when perform DELETE or TRUNCATE properly.
  - Fixed Cache_METADATA_TABLES insert into METADATA_PRIVILEGES duplicate key.
  - Redo how the gateKeeper/gateKeeperCheck procedures detect cluster processing and wait for all nodes to complete.
    - Potential issue: If there is a cluster break and the nodes sub-partition and then there may be different number of nodes in each sub-partition which will surely throw off the computations in the gateKeeperCheck procedure.
  - Published new resources:
    - /services/databases/ASAssets/KPImetrics/metadata/reportMetadataAllCount
    - /services/databases/ASAssets/KPImetrics/metadata/reportMetadataAllCountArch
    - /services/databases/ASAssets/KPImetrics/metrics/metrics_requests_groupby_date
    - /services/databases/ASAssets/KPImetrics/metrics/metrics_requests_groupby_nodehost_nodeport
    - /services/databases/ASAssets/KPImetrics/metrics/metrics_resources_usage_groupby_date
    - /services/databases/ASAssets/KPImetrics/metrics/metrics_resources_usage_groupby_nodehost_nodeport
    - /services/databases/ASAssets/KPImetrics/metrics/metrics_sessions_groupby_date
    - /services/databases/ASAssets/KPImetrics/metrics/metrics_sessions_groupby_nodehost_nodeport

© Copyright 2000-2017 TIBCO Software Inc.

- Upgrade Changes:
  - modify_LDAP_PERSON - Added a list of columns to be removed from LDAP_PERSON in order to simplify and reduce overhead. See removeColumnList.
  - 04_pqCreateDrop_KPI_Tables_sqlserver_kpi2_tables - Create multiple tables for METRICS_ALL_RESOURCES, METRICS_ALL_USERS and LDAP_PERSON that will be used for loading and access. 1) Keep tables accessible all the time while loading new data. 2) Increase parallel NODE load and efficiency by removing the gatekeeper.

Release 2020Q200 [Mar 12 2020]

- Release 2020Q200
  - Includes 2020.102 + Requires Utilities 2020Q200 to synchronize with migrated utilities required by KPImetrics.
  - Installation – MOVE OR UPDATED THE FOLLOWING TO Utilities 2020Q200
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/freeUnusedMemory
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getBasicResourceCursor_PROCEDURE_CURSOR
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getBasicResourceCursor_SQL_TABLE
    - /shared/ASAssets/KPImetrics_installation/Helpers/Utilities/copyResource
    - /shared/ASAssets/KPImetrics_installation/Helpers/Utilities/getConstantV2
    - /shared/ASAssets/KPImetrics_installation/Helpers/Utilities/getTrigger
    - /shared/ASAssets/KPImetrics_installation/Helpers/Utilities/updateResourceEnabled
    - /shared/ASAssets/KPImetrics_installation/Helpers/Utilities/updateUser
  - Cache_METADATA_TABLES - Updated to use Utilities 2020Q200 instead of KPImetrics local version of Utilities mentioned below.
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/freeUnusedMemory
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getBasicResourceCursor_PROCEDURE_CURSOR
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getBasicResourceCursor_SQL_TABLE
- Includes patch 2020Q102 – Modified to alleviate SQL Server blocking
  - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/00_ExecuteDMLSelect - generic select and response
  - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/00_ExecuteDMLSelect - generic select and response
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pStartWorkflow - use getTableCount()
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/gateKeeperCheck - use getTableCount()

© Copyright 2000-2017 TIBCO Software Inc.

- /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getBasicResourceCursor_PROCEDURE_CURSOR
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getBasicResourceCursor_SQL_TABLE
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getResourceAttributesNormalized
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/executeMetricsSqlControl
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pUpdateEventRegLog
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/verifyProcessRunningMetricsJobDetails
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METADATA_TABLES
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_DELETE_COLLECTION_exec
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pValidateUpdateStatusTables
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckDBMSSchedulerError
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pRebuildIndexes
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pValidateUpdateStatusTables
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckCISWorkflowStatusFail
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckMetricsActivity
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pPurgeData
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_RESOURCES
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_USERS
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_AccessByUserOverTime
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ActiveResourcesOverPeriodOfTime
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_CIS_SYSTEM_RESOURCES
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_DISK
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_IO
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_MEMORY

© Copyright 2000-2017 TIBCO Software Inc.

- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_DATASOURCES
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC_ADHOC
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_RESOURCE_LINEAGE
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ALL_RESOURCES
  - fixed bug for RESOURCE_NAME, PARENT_PATH, RESOURCE_PATH where name contains a / and requires double quotes around name.
- /shared/ASAssets/KPImetrics/Business/Business/requests/pExceededMemoryPercentRequests
- /shared/ASAssets/KPImetrics/Business/Business/requests/pLongRunningRequests

Release 2020Q101 [Feb 25 2020]

- General release goals
  - KPI Metadata improvements. Purge concurrency improvements.
  - Redefine Strategy - simplify processing
    - Dropped table METRICS_ALL_RESOURCES_STG - no longer needed for staging ALL_RESOURCES data.
    - METRICS_ALL_RESOURCES will no longer maintaining a resource history. Only 1 copy of the current ALL_RESOURCES will be cached at any one time.
    - METRICS_ALL_USERS will no longer maintain user history. Only 1 copy of the current ALL_USERS will be cached at any one time.
    - METRICS_LDAP_PERSON will no longer maintain ldap history. Only 1 copy of the current ldap users will be cached at any one time.
    - Alternative strategy - Cache_METADATA_TABLES will allow the configuration and archival of ALL_RESOURCES and ALL_USERS over time.
    - Published resource changes:
      - KPImetrics.resource.vAllResourcesMax - dropped
      - KPImetrics.resource.vAllResourcesHist - renamed to vAllResources - retrieve current ALL_RESOURCES
      - KPImetrics.resource.METRICS_ALL_RESOURCES_groupby_nodehost_nodeport - renamed to vAllResources_GroupBy_NodehostNodeport
      - KPImetrics.users.vAllUsersMax - dropped
      - KPImetrics.users.vAllUsersHist - renamed to vAllUsers - retrieve current ALL_USERS

© Copyright 2000-2017 TIBCO Software Inc.

- KPI Metrics:
  - Added a new table METRICS_JOB_DELETE_CHECK - status check for the TDV metrics delete.
  - Added new procedure - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMetricsDeleteCheck
  - Added new trigger - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_35_DBMSDeleteCheck
  - ALTER TABLE "{0}"."METRICS_ACR_RCD" RENAME COLUMN "loadtime" TO "loaddate"
  - ALTER TABLE "{0}"."METRICS_ACR_RCT" RENAME COLUMN "loadtime" TO "loaddate"
  - ALTER TABLE "{0}"."METRICS_LDAP_PERSON" RENAME COLUMN "loadtime" TO "loaddate"
  - ALTER TABLE "{0}"."METRICS_ALL_USERS" RENAME COLUMN "LOAD_TIME" TO "LOAD_DATE"
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" RENAME COLUMN "LOAD_TIME" TO "LOAD_DATE"
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" RENAME COLUMN "RESOURCEPATH" TO "RESOURCE_PATH"
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" RENAME COLUMN "OWNER" TO "OWNER_NAME"
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" ADD "RESOURCE_LINK_TYPE" VARCHAR2(40)
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" ADD "OWNER_DOMAIN" VARCHAR2(255)
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" ADD "ANNOTATION" CLOB
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" MODIFY "RESOURCE_TYPE" VARCHAR2(40)
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" MODIFY "RESOURCE_SUBTYPE" VARCHAR2(40)
  - ALTER TABLE "{0}"."METRICS_ALL_RESOURCES" MODIFY "GUID" VARCHAR2(255)
- KPI Metadata:
  - Added a new table METADATA_ALL_RESOURCES - improve processing performance. cache from existing METRICS_ALL_RESOURCES.
  - Added a new table METADATA_ALL_PRIVILEGES - improve processing performance.
  - Added a new table METADATA_ALL_USERS_GROUPS[_ARCH] - improve processing performance.
  - Added a new table METADATA_PRIVILEGE_USER[_ARCH] - provides a many-to-many between METADATA_PRIVILEGE and METADATA_ALL_USER_GROUPS.
  - Added column ANNOTATION to METADATA_RESOURCE[_ARCH]
  - Added column ANNOTATION to METADATA_RESOURCE_COLUMN[_ARCH]
  - Added column ANNOTATION to METADATA_DATASOURCE[_ARCH]
  - Added column PRIVILEGE_ID, LAYER_TYPE, NAME_ID to METADATA_PRIVILEGE[_ARCH]
  - Added column MESSAGE to METADATA_POLICY_ASSIGNMENT - when a policy assignment points to a resource path that no longer exists, the log the message.

© Copyright 2000-2017 TIBCO Software Inc.

- Added column ASSIGN_PRIVILEGES to METADATA_CONST_LAYERS[_ARCH]
- Dropped column USER_NAME in METADATA_PRIVILEGE[_ARCH] in favor of the join table METADATA_PRIVILEGE_USER
- Cache_METADATA_TABLES - modified to use more efficient processing of ALL_RESOURCES and ALL_PRIVILEGES. Modified for column changes.

- Default Triggers
  - Defaulted kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST to OFF due to intensive processing. Customer may turn on if required.
  - Defaulted kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE to OFF due to intensive processing. Customer may turn on if required.
  - Defaulted kpimetricsTrig_33_DeleteCollection to OFF due to blocking issues with SQL Server. Customer should not turn this on.
- Override Triggers
  - kpimetricsTrig_16_PurgeHistoryData - override schedule to once a week SAT at 2:30 AM
  - kpimetricsTrig_34_DBMSRebuildIndexes - override schedule to once a week SUN at 3:00 AM
  - kpimetricsTrig_01_Cache_ALL_RESOURCES - override schedule to every 2 hours at 1:00 AM
  - kpimetricsTrig_02_Cache_ALL_USERS - override schedule to every 2 hours at 1:00 AM
  - kpimetricsTrig_30_DBMSScheduler - override schedule to every 2 hours at 1:30 AM (coincide with 30 min later than kpimetricsTrig_01 and kpimetricsTrig_02)
  - kpimetricsTrig_31_DBMSSchedulerError - override schedule to every 2 hours at 2:00 AM (coincide with 30 min later than kpimetricsTrig_30)
  - kpimetricsTrig_40_Cache_METADATA_TABLES - override schedule to every 1 day at 11:30 PM (coincide to run 30 min later than kpimetricsTrig_01_Cache_ALL_RESOURCES because it uses its cached data)
- New Triggers / Procedure
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_50_MetricsPersistenceFailure
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_51_MetricsTruncateFailure
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_52_MetricsBackupFailure
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_53_MetricsRestoreFailure
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMetricsSystemEvent
  - Updated commonValues.eventRegistrationTypeList with new event 'METRICS_FAILURE'

© Copyright 2000-2017 TIBCO Software Inc.

        - Updated /shared/ASAssets/KPImetrics/Customize/pqInsert_METRICS_EVENT_REGISTRATION with documentation on new event 'METRICS_FAILURE'
        - Updated /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pGetEmailSubscriptions
        - Updated /shared/ASAssets/KPImetrics/Configuration/pMetricsEventRegistrationList
        - Updated /shared/ASAssets/KPImetrics/Configuration/pMetricsEventRegistrationSubscribe
        - Updated /shared/ASAssets/KPImetrics/Configuration/pMetricsEventRegistrationUnsubscribe
        - Updated /shared/ASAssets/KPImetrics/Customize/defaultTriggersToEnable
    - Modified pCheckDBMSSchedulerError to put a tracking row in METRICS_JOB_DETAILS instead of using pStartWorkflow/pEndWorkflow.
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckDBMSSchedulerError
    - Purge Data (other than the 3 core metrics tables and METADATA tables)
        - commonValues.EX_SUSPENDED - added a public exception that gets used by the following:
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pStartWorkflow
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_RESOURCES
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_USERS
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_AccessByUserOverTime
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ActiveResourcesOverPeriodOfTime
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ResourceCount_Details
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ResourceCount_Total
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_CIS_SYSTEM_RESOURCES
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_CPU_MEMORY_CHECKER
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LDAP_PERSON
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_DISK
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_IO
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_MEMORY
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC_ADHOC
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS

© Copyright 2000-2017 TIBCO Software Inc.

- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_RESOURCE_LINEAGE
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_CACHES
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_DATASOURCES
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckCISWorkflowStatusFail
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckExceedMemoryPercentRequests
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckLongRunningRequests
- /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pCheckMetricsActivity

Release 2020Q100 [Jan 23 2020]

- General release goals
  - Created a data dictionary document - KPImetrics Data Dictionary v1.1.pdf
  - Changed delete strategy for P_METRICS_ALL_TABLES
    - For both Oracle and SQL Server - do not delete collection tables from KPImetrics. Only DV metrics will control this.
    - Allow DV metrics to delete only to eliminate SQL Server concurrency issue with simultaneous INSERTS and DELETES.
    - Note: Oracle does not have this issue but modified with same behavior for consistency in strategies.
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec
    - /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction/P_METRICS_ALL_TABLES
    - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/P_METRICS_ALL_TABLES
    - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/P_METRICS_ALL_TABLES
    - /shared/ASAssets/KPImetrics_installation/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_700
    - /shared/ASAssets/KPImetrics_installation/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_800
    - /shared/ASAssets/KPImetrics/Configuration/updateTriggers - Modified to force kpimetricsTrig_33_DeleteCollection to be OFF.
  - Added transaction blocks around database SQL statements
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pPARTITION_MANAGER_exec
    - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/pqPartition_metrics_history_tables_ADD
    - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/pqPartition_metrics_history_tables_DROP
    - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/pqPartition_metrics_history_tables_ADD

© Copyright 2000-2017 TIBCO Software Inc.

        - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/pqPartition_metrics_history_tables_DROP
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pUpdateEventRegLog
    - Added transaction blocks around database SQL statements
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pPARTITION_MANAGER_exec
        - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/pqPartition_metrics_history_tables_ADD
        - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/pqPartition_metrics_history_tables_DROP
        - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/pqPartition_metrics_history_tables_ADD
        - /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/pqPartition_metrics_history_tables_DROP
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pUpdateEventRegLog
    - Documentation
        - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_DATASOURCES
    - Installation scripts
        - /shared/ASAssets/KPImetrics_installation/3_Post_Upgrade
        - /shared/ASAssets/KPImetrics_installation/backupCustomResources - Save trigger metdata
        - /shared/ASAssets/KPImetrics_installation/Helpers/General/getTrigger
        - /shared/ASAssets/KPImetrics_installation/Helpers/General/getTriggerMetadata
        - /shared/ASAssets/KPImetrics_installation/Helpers/General/turnOffAllTriggers
    - Missing Column
        - ../Physical/Formatting/METRICS_EVENT_REGISTRATION added EXCLUDE_TEXT
        - ../Business/Business/requests/vMetricsSqlRequestUniqueSqlTemplatesByUser added "domain"
        - ../Business/Business/requests/vMetricsSqlRequestUniqueSqlTemplatesByUserByDate - added "domain"
        - ../Business/Business/requests/vMetricsSqlRequestUniqueSqlTemplatesClobByUser - added "domain"
        - ../Business/Business/requests/vMetricsSqlRequestUniqueSqlTemplatesClobByUserByDate - added "domain"
    - Modified column names to lower case style
        - /shared/ASAssets/KPImetrics/Application/workflow/vJobDetailsStatusSummary
    - Moved to KPImetricsAdmin catalog
        - /services/databases/ASAssets/KPImetrics/configuration/updateTriggers

Release 2019Q402 [Dec 23 2019]

- General release goals
    - Improved SQL Server database stability
    - Improved script execution for a cluster to reduce deadlocks
    - Metrics data transfer procedure batching changes SQL Server
    - Better debug and error handling
    - Moved all /DDL procedures to KPImetrics_installation and out of KPImetrics

© Copyright 2000-2017 TIBCO Software Inc.

  - Revamped the upgrade scripts to be easier to execute
  - Refactored customized resources to a single folder /KPImetrics/Customize
- Refactored Code
  - Moved /KPImetrics/Configuration/commonValues to /KPImetrics/Customize/commonValues
    - added dataTransferBatchDeleteMax
    - allowed SQL Server to use dataTransferBatchInsertMax and dataTransferBatchUpdateMax are used by SQL Server.
  - Moved /KPImetrics/Configuration/defaultTriggersToEnable to /KPImetrics/Customize/defaultTriggersToEnable
    - Added trigger kpimetricsTrig_25_ExecuteMetricsSqlControl
  - Moved /KPImetrics/Physical/Metadata/System/LDAP_PERSON to /KPImetrics/Customize/LDAP_PERSON
  - Moved all DDL scripts to /shared/ASAssets/KPImetrics_installation folder except the following:
  - Moved /DDL/Common/07_pqInsert_KPI_Tables_METRICS_JOB_tables to /Customize/pqInsert_METRICS_JOB_tables and modified the structure.
  - Moved /DDL/Common/08_pqInsert_KPI_Tables_METRICS_EVENT_REGISTRATION to /Customize/pqInsert_METRICS_EVENT_REGISTRATION
  - Moved /DDL/Common/10_pqInsert_Metadata_Tables_METADATA_Constants to /Customize/pqInsert_METADATA_Constants
  - Moved /DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_DROP to /KPI_oracle/pqPartition_metrics_history_tables_DROP
  - Moved /DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_ADD to /KPI_oracle/pqPartition_metrics_history_tables_ADD
  - Moved /DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_ROW_DISTRIBUTION to /KPI_oracle/pqPartition_metrics_history_tables_ROW_DISTRIBUTION
  - Moved /DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_DROP to /KPI_sqlserver/pqPartition_metrics_history_tables_DROP
  - Moved /DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_ADD to /KPI_sqlserver/pqPartition_metrics_history_tables_ADD
  - Moved /DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_ROW_DISTRIBUTION to /KPI_sqlserver/pqPartition_metrics_history_tables_ROW_DISTRIBUTION
- Modified Metadata DDL scripts
  - /shared/ASAssets/KPImetrics_installation/DDL/Oracle/04_pqCreateDrop_KPI_Tables_oracle_kpi_tables
  - /shared/ASAssets/KPImetrics_installation/DDL/SqlServer/04_pqCreateDrop_KPI_Tables_sqlserver_kpi_tables
  - /shared/ASAssets/KPImetrics_installation/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_700
  - /shared/ASAssets/KPImetrics_installation/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_800
- Added DDL Upgrade scripts
  - /shared/ASAssets/KPImetrics_installation/DDL/Oracle/Upgrade/upgrade_2019Q401_to_2019Q402_pqCreateDrop_Tables_oracle
  - /shared/ASAssets/KPImetrics_installation/DDL/SqlServer/Upgrade/upgrade_2019Q401_to_2019Q402_pqCreateDrop_Tables_sqlserver
  - /shared/ASAssets/KPImetrics_installation/DDL/UpgradeResources/upgrade_2019Q401_to_2019Q402_07InsertJobTables
  - /shared/ASAssets/KPImetrics_installation/DDL/UpgradeResources/upgrade_2019Q401_to_2019Q402_commonValues
  - /shared/ASAssets/KPImetrics_installationDDL/UpgradeResources/upgrade_2019Q401_to_2019Q402_defaultTriggersToEnable

© Copyright 2000-2017 TIBCO Software Inc.

- Error Handling output to log
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pStartWorkflow
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/pEndWorkflow
  - All procedures in /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache
- Added new resources
  - /services/databases/ASAssets/KPImetrics/systemUsage/vDatasourceCurrentStatusChanges
  - /shared/ASAssets/KPImetrics/Application/systemUsage/vDatasourceCurrentStatusChanges
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_25_ExecuteMetricsSqlControl
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pExecuteMetricsSqlControl
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/executeMetricsSqlControl
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getServerNames
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getServerNamesXSLT
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/verifyProcessRunningMetricsJobDetails
  - /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction/METRICS_HISTORY_TABLES_ROW_DISTRIBUTION
  - /shared/ASAssets/KPImetrics/Physical/Formatting/METRICS_HISTORY_TABLES_ROW_DISTRIBUTION
  - /shared/ASAssets/KPImetrics/Business/Logical/metrics/metrics_history_tables_row_distribution
  - /shared/ASAssets/KPImetrics/Application/metrics/metrics_history_tables_row_distribution
  - /services/databases/ASAssets/KPImetrics/metrics/metrics_history_tables_row_distribution
- Added new database columns and modified Formatting layer views to bring in new column
  - METRICS_CIS_WORKFLOW: ERROR_MESSASGE, WORKFLOW_START_TIME_STR
    - Support better error handling and updating. Allows the entire error trace to be saved into ERROR_MESSAGE.
  - METRICS_JOB_FILTERS: RESOURCE_PATH, RESOURCE_TYPE, PATH_TYPE
    - Support filtering based on the resource path and type
  - METRICS_JOB_DETAILS: REQ_END_DATE, CORRELATION_ID, PROCESSED_NODE_HOST, PROCESSED_NODE_PORT, PROCESSED_START_DATE, ERROR_MESSAGE
- Modified
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pRebuildIndexes
    - Synchronized execution with pMETRICS_ALL_TABLES_exec and pMETRICS_DELETE_COLLECTION_exec
    - For Oracle, dynamic rebuild based on what is in Oracle catalogs
  - shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec
    - Synchronized execution with pRebuildIndexes and pMETRICS_DELETE_COLLECTION_exec
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_DELETE_COLLECTION_exec
    - Synchronized execution with pMETRICS_ALL_TABLES_exec and pRebuildIndexes
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_DATASOURCES
    - Removed dataSourceType and dataSourceSubtype as it is not used.
    - Fixed the logic to only test a datasource once.
    - Added logic to insert a record for a deleted datasource instead of updating the most recent one.
  - /shared/ASAssets/KPImetrics/Configuration/pMaintenanceUpdateHostnamePort
- Modified Input Parameters

© Copyright 2000-2017 TIBCO Software Inc.

- /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction/P_METRICS_ALL_TABLES
- /shared/ASAssets/KPImetrics/Physical/Physical/Abstraction/P_METRICS_DELETE_COLLECTION
- /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/P_METRICS_ALL_TABLES
- /shared/ASAssets/KPImetrics/Physical/Physical/KPI_oracle/P_METRICS_DELETE_COLLECTION
- /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/P_METRICS_ALL_TABLES
- /shared/ASAssets/KPImetrics/Physical/Physical/KPI_sqlserver/P_METRICS_DELETE_COLLECTION

- Modified to remove the projection and put in a "SELECT *"
  - /shared/ASAssets/KPImetrics/Business/Logical/metrics/metrics_requests
  - /shared/ASAssets/KPImetrics/Business/Logical/metrics/metrics_resources_usage
  - /shared/ASAssets/KPImetrics/Business/Logical/metrics/metrics_sessions
- Removed dataSourceType and dataSourceSubtype as it is not used:
  - /shared/ASAssets/KPImetrics_installation/2_Post_Installation
  - /shared/ASAssets/KPImetrics/Business/Logical/metrics/metrics_all_kpimetrics_table_counts
  - /shared/ASAssets/KPImetrics/Configuration/rebindPhysicalAbstraction
  - /shared/ASAssets/KPImetrics/Configuration/rebindPhysicalDatabaseType
  - /shared/ASAssets/KPImetrics/Configuration/updateTriggers
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_RESOURCES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_ALL_USERS
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_AccessByUserOverTime
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ActiveResourcesOverPeriodOfTime
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ResourceCount_Details
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_AllCustom_ResourceCount_Total
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_CIS_SYSTEM_RESOURCES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_CPU_MEMORY_CHECKER
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LDAP_PERSON
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_DISK
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_IO
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_LOG_MEMORY
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METADATA_TABLES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC_ADHOC
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_REQUEST_EXEC_REPROCESS
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METRICS_SQL_RESOURCE_LINEAGE
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_CACHES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_SYS_DATASOURCES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pPARTITION_MANAGER_exec
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pRebuildIndexes
  - All resources in /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle
  - All resource in /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer

© Copyright 2000-2017 TIBCO Software Inc.

- Debug param added to procedure input
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_00_CheckMetricsActivityDebug
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_01_Cache_ALL_RESOURCES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_02_Cache_ALL_USERS
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_03_Cache_LDAP_PERSON
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_04_Cache_CIS_SYSTEM_RESOURCES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_05_Cache_CPU_MEMORY_CHECKER
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_06_Cache_LOG_DISK
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_07_Cache_LOG_IO
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_08_Cache_LOG_MEMORY
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_10_Cache_METRICS_SQL_REQUEST
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_11_Cache_METRICS_SQL_REQUEST_REPROCESS
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_12_Cache_SYS_CACHES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_13_Cache_SYS_DATASOURCES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_14_CheckCISWorkflowStatusFail
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_15_CheckMetricsActivity
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_16_PurgeHistoryData
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_17_CheckExceedMemoryPercentRequests
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_18_CheckLongRunningRequests
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_19_AllCustom_AccessByUserOverTime
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_20_AllCustom_ActiveResourcesOverPeriodOfTime
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_21_AllCustom_ResourceCount_Details
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_22_AllCustom_ResourceCount_Total
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_30_DBMSScheduler
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_31_DBMSSchedulerError
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeTriggers/kpimetricsTrig_33_DeleteCollection

Release 2019Q401 [Nov 6 2019]

- Modified Metadata DDL scripts
  - Added LINEAGE_ORDER to METADATA_NON_COMPLIANT
  - Modified column METADATA_PRIVILEGE.USER_NAME VARCHAR(255)
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/10_pqCreateDrop_Metadata_Tables_oracle_metadata_tables
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/10_pqCreateDrop_Metadata_Tables_sqlserver_metadata_tables
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Common/10_pqInsert_Metadata_Tables_METADATA_Constants - Documentation/formatting changes

© Copyright 2000-2017 TIBCO Software Inc.

  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/Upgrade/upgrade_2019Q400_to_2019Q401_pqCreateDrop_Tables_oracle
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/Upgrade/upgrade_2019Q400_to_2019Q401_pqCreateDrop_Tables_sqlserver
- Consolidated upgrade scripts to simplify upgrade process
  - Consolidated upgrade_8000Q000_pqCreateDrop_Final_[oracle|sqlserver] into upgrade_2019Q101_to_2019Q102_pqCreateDrop_Tables_[oracle|sqlserver]
  - Consolidated upgrade_9000Q000_pqUpdate_METRICS_ALL_RESOURCES into upgrade_2018Q103_to_2018Q300_pqCreateDrop_Tables_[oracle|sqlserver]
  - Consolidated upgrade_9001Q000_pqUpdate_metrics_hist_tables into upgrade_2018Q401_to_2019Q100_pqCreateDrop_Tables_[oracle|sqlserver]
- Fixed Datasource Lineage
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/Cache_METADATA_TABLES
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getDatasourceDependencyMetadata - new
- Created a new Resource/Datasource Lineage report
  - /shared/ASAssets/KPImetrics/Application/metadata/reportResourceDatasourceLineage - new report
  - /services/databases/ASAssets/KPImetrics/metadata/reportResourceDatasourceLineage
- Modified Non-Compliant report
  - /shared/ASAssets/KPImetrics/Application/metadata/reportMetadataNonCompliantLayers
- New column - LINEAGE_ORDER
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_[11g|12c]/CIS_KPI/METADATA_NON_COMPLIANT
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_[2012|2014]/CIS_KPI/dbo/METADATA_NON_COMPLIANT_ARCH
  - /shared/ASAssets/KPImetrics/Physical/Formatting/METADATA_NON_COMPLIANT
  - /shared/ASAssets/KPImetrics/Business/Logical/metadata/vMetadataNonCompliant
  - /shared/ASAssets/KPImetrics/Business/Business/metadata/vMetadataNonCompliant
  - /shared/ASAssets/KPImetrics/Application/metadata/vMetadataNonCompliant
  - /services/databases/ASAssets/KPImetrics/metadata/vMetadataNonCompliant

  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_[11g|12c]/CIS_KPI/METADATA_NON_COMPLIANT
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_[2012|2014]/CIS_KPI/dbo/METADATA_NON_COMPLIANT_ARCH
  - /shared/ASAssets/KPImetrics/Physical/Formatting/METADATA_NON_COMPLIANT_ARCH

© Copyright 2000-2017 TIBCO Software Inc.

- /shared/ASAssets/KPImetrics/Business/Logical/metadata/vMetadataNonCompliantArch
- /shared/ASAssets/KPImetrics/Application/metadata/vMetadataNonCompliantArch
- /shared/ASAssets/KPImetrics/Business/Business/metadata/vMetadataNonCompliantArch
- /services/databases/ASAssets/KPImetrics/metadata/vMetadataNonCompliantArch

- Fixed missing columns - LINEAGE_ORDER and MESSAGE
  - /shared/ASAssets/KPImetrics/Physical/Formatting/METADATA_RESOURCE_LINEAGE - missing columns
  - /shared/ASAssets/KPImetrics/Business/Logical/metadata/vMetadataResourceLineage
  - /shared/ASAssets/KPImetrics/Business/Business/metadata/vMetadataResourceLineage
  - /shared/ASAssets/KPImetrics/Application/metadata/vMetadataResourceLineage
  - /services/databases/ASAssets/KPImetrics/metadata/vMetadataResourceLineage

  - /shared/ASAssets/KPImetrics/Physical/Formatting/METADATA_RESOURCE_LINEAGE_ARCH - missing columns
  - /shared/ASAssets/KPImetrics/Business/Logical/metadata/vMetadataResourceLineageArch
  - /shared/ASAssets/KPImetrics/Business/Business/metadata/vMetadataResourceLineageArch
  - /shared/ASAssets/KPImetrics/Application/metadata/vMetadataResourceLineageArch
  - /services/databases/ASAssets/KPImetrics/metadata/vMetadataResourceLineageArch
- Modified column - METADATA_PRIVILEGE.USER_NAME VARCHAR(255)
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_[11g|12c]/CIS_KPI/METADATA_PRIVILEGE
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_[2012|2014]/CIS_KPI/dbo/METADATA_PRIVILEGE_ARCH
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_oracle_[11g|12c]/CIS_KPI/METADATA_PRIVILEGE
  - /shared/ASAssets/KPImetrics/Physical/Metadata/KPI_sqlserver_[2012|2014]/CIS_KPI/dbo/METADATA_PRIVILEGE_ARCH

Release 2019Q400 [Oct 24 2019]

- SQL Server Schema qualifer in DDL scripts. Oracle DDL modifications.
- Fixed bugs
  - /shared/ASAssets/KPImetrics_installation/Helpers/createTables
- Modified documentation header
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_700
  - shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_800
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_ADD - modified algorithm to add partitions from last one to current date+1.

© Copyright 2000-2017 TIBCO Software Inc.

  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_DROP

- Added schema qualifier to SQL Server Scripts
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables_700
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables_800
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_700
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_800
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_ADD - modified algorithm to add partitions from last one to current date+1.
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_DROP
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/04_pqCreateDrop_KPI_Tables_sqlserver_kpi_tables
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/10_pqCreateDrop_Metadata_Tables_sqlserver_metadata_tables

- Modified existing DDL by adding additional where clause into the delete statements.
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_700
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_800
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_700
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_800

Release 2019Q301 [Aug 31 2019]

- Added new feature for capturing metadata [kpiMetadata].
  - Added several new tables startig with METADATA_<tablename>.
  - Added /metadata in all layers including a published schema called metadata.
  - Added Cache_METADATA_TABLES and kpimetricsTrig_40_Cache_METADATA_TABLES to process.
  - Added 10_pqCreateDrop_Metadata_Tables_oracle_metadata_tables
  - Added 10_pqCreateDrop_Metadata_Tables_sqlserver_metadata_tables
  - Added 10_pqInsert_Metadata_Tables_METADATA_Constants
  - Added upgrade script upgrade_2019Q200_to_2019Q301_pqCreateDrop_Tables_oracle
  - Added upgrade script upgrade_2019Q200_to_2019Q301_pqCreateDrop_Tables_sqlserver
  - Created new documentation
- Fixed various bugs. Added purge history rule for no partitioning use case. Added a DDL upgrade script for history tables.
  - Added "purgeHistory" rule for no partitioning which is used by pMETRICS_ALL_TABLES_exec.
    - /shared/ASAssets/KPImetrics/Configuration/commonValues
  - Added the ability to delete history rows when there is no partitioning configured. partitionNumber=0.
  - Prior to this, partitioning was required in order to drop partitions to delete history data.
    - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pMETRICS_ALL_TABLES_exec

© Copyright 2000-2017 TIBCO Software Inc.

- Modified validation of METRICS_JOB_DETAILS by using jobTableList to search for various JOB_TABLE_NAME variations.
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pValidateUpdateStatusTables
- Modified pPurgeData to add the missing METRICS_JOD_DETAILS to purgeWorkflowData rule
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pPurgeData
- Modified to pRebuildIndexes to invoke getVersion resolving an issue when the version is 8.x
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/pRebuildIndexes
- Modified commonValues and removed all custom functions.
- Modified each procedure that invokes that commonValues variable to call the explicit path of the custom function.
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/CPU_MEMORY_CHECKER
  - /shared/ASAssets/KPImetrics/Business/Business/requests/pExceededMemoryPercentRequests
  - /shared/ASAssets/KPImetrics/Business/Business/requests/pLongRunningRequests
  - /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/sqlParserV1_2/utility/parseColumnExpression
- Added DDL upgrade script:
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/Upgrade/upgrade_9001Q000_pqUpdate_metrics_hist_tables
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/Upgrade/upgrade_9001Q000_pqUpdate_metrics_hist_tables
- Modified existing DDL:
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/Upgrade/upgrade_8000Q000_pqCreateDrop_Final_oracle - added input executeDDL
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/Upgrade/upgrade_9000Q000_pqUpdate_METRICS_ALL_RESOURCES - added input executeDDL
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_700 - ensure null or empty string userKeyTransform and domainKeyTransform get default values
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/Oracle/06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_800 - ensure null or empty string userKeyTransform and domainKeyTransform get default values
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/Upgrade/upgrade_8000Q000_pqCreateDrop_Final_sqlserver - added input executeDDL
  - /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/Upgrade/upgrade_9000Q000_pqUpdate_METRICS_ALL_RESOURCES - added input executeDDL

© Copyright 2000-2017 TIBCO Software Inc.

- /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_700 - fixed bug with replace: REPLACE(sqlRow.sqlStatement, 'ALL TO ([{HISTORY}])', 'ALL TO ([PRIMARY])');
- /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_800 - fixed bug with replace: REPLACE(sqlRow.sqlStatement, 'ALL TO ([{HISTORY}])', 'ALL TO ([PRIMARY])');
- /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_ADD - fixed bug with replace: REPLACE(sqlRow.sqlStatement, 'NEXT USED [{HISTORY}]', 'NEXT USED [PRIMARY]');
- /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_700 - ensure null or empty string userKeyTransform and domainKeyTransform get default values
- /shared/ASAssets/KPImetrics/Physical/Metadata/DDL/SqlServer/06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_800 - ensure null or empty string userKeyTransform and domainKeyTransform get default values.

Release 2019Q300 [Aug 5 2019]

- Release Objective: Bug fixes.
  - Fixed bug where pRebuildIndexes fails. Fixed with admin interface invocation.
    - Indexes do not start rebuilding until all processare are inactive.
      - For METRICS_CIS_WORKFLOW, WORKFLOW_STATUS <> 'I'
      - For METRICS_JOB_DETAILS, STATUS <> 'BEGIN JOB'
  - The following two new resources act in conjunction with pRebuildIndexes. They are used to ensure that the METRICS_CIS_WORKFLOW and METRICS_JOB_DETAILS do not have in progress status when there are no active processes. It is the responsibility of each node in a cluster to manage its own status records.
    - For METRICS_CIS_WORKFLOW where WORKFLOW_STATUS='I' and there are no active processes, update WORKFLOW_STATUS='F' and set MESSAGE='Failure for unknown reason.'.
    - For METRICS_JOB_DETAILS where STATUS='BEGIN JOB' and there are no active processes, update STATUS='FAILURE' and set ADDITIONAL_INFO='Failure for unknown reason.'.
  - Added new trigger /ClusterSafeTriggers/kpimetricsTrig_24_ValidateUpdateStatusTables
  - Added new procedure /ClusterSafeCache/pValidateUpdateStatusTables
  - Modified defaultTriggersToEnable to add the new trigger "kpimetricsTrig_24_ValidateUpdateStatusTables" to the list.

Release 2019Q200 [Jun 13 2019]

- Release Objective: Migrated some basic scripts to the Utilities.
  - Requires Utilities 2019.200 version
  - Migrated getCodedString to the Utilities

© Copyright 2000-2017 TIBCO Software Inc.

- Migrated encodedValues to the Utilities
- Migrated findOpenClosePair to the Utilities
- Rebound extractSQL to numOccurrences in Utilities
- Rebound parseArithmeticOperators to numOccurrences in Utilities
- Rebound parseSqlScriptComplex to numOccurrences in Utilities
- Rebound to getCurrentTimestamp in Utilities.
- Removed /shared/ASAssets/KPImetrics/Physical/Metadata/System/Helpers/getCurrentTimestamp.
- pPurgeData - Put explicit BEGIN blocks around each DELETE statement in order to fix a blocking condition with METRICS_CIS_WORKFLOW.
- Modified METRICS_SYS_CACHES.STATUS from VARCHAR(20) to VARCHAR(24) for TDV 8.1
- sqlParserV1_2 - Converted all VECTORS to XML as VECTOR parameters has been associated with memory leaks.

Release 2019Q102 [Mar 12 2019]

- Release Objective: Added upgrade scripts. Added “group” to metrics_requests and metrics_sessions for 7.0 to bring up to par with 8.0.
  - Added upgrade scripts starting with 2018Q103 for oracle and sqlserver.
    - Allows upgrading from 2018Q103 or higher on 7.x to 7.x.
    - Allows upgrading from 2018Q103 or higher on 7.x to 8.x.
  - Added indexes for METRICS_CIS_WORKFLOW:
    - METRICS_CIS_WORKFLOW_IDX1 and METRICS_CIS_WORKFLOW_IDX2
  - Modified schemas [collection, staging] for 7.0.
    - "group" was added to metrics_sessions and metrics_requests for 7.0.
  - Modified schemas [history] to work with 8.0 modified metrics schemas.
    - "group" was added to metrics_sessions_hist and metrics_requests_hist for 7.0 as "user" and "domain" already exists.
  - Modified Oracle DDL Scripts:
    - Modified 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables_700 to add "group"
    - Modified 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_700 to add "group"
    - Modified 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_700 to add "group"
  - Modified SQL Server DDL Scripts:
    - Modified 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables_700 to add "group"
    - Modified 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_700 to add "group"

© Copyright 2000-2017 TIBCO Software Inc.

        - Modified 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_700 to add "group"
    - Renamed updateImpacteResources to updateImpactedResources (missing d in Impacted).
    - Modified /shared/ASAssets/KPImetrics_installation to account for updateImpactedResources.
    - Modified all the scripts in /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache that contained "INSERT INTO" statements to specify INSERT and SELECT columns and not use SELECT *.

Release 2019Q101 [Jan 30 2019]

- Release Objective: Added support for DV 8.0 as the out-of-the-box metrics tables [metrics_requests and metrics_sessions] added user, domain and group. KPI installation automatically checks for DV version 7.0 or 8.0 and executes the appropriate scripts to install KPImetrics.
    - Bug Fixes:
        - sqlParserV1_2 - Various scripts. Fixed a bug when server IGNORE_TRAILING_SPACES=true. Changed INSTR functions with a trailing space to use SELECT {OPTION IGNORE_TRAILING_SPACES="FALSE"} INSTR(sqlScript, ' ')
    - Modified KPImetrics_installation scripts to install for 7.0 or 8.0 automatically.
    - Modified schemas [collection, staging, history] to work with 8.0 modified metrics schemas. The columns "user", "domain", "group" were added to metrics_sessions and metrics_requests for 8.0.
    - Modified Oracle DDL Scripts:
        - Renamed 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables to 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables_700
        - Added 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables_800
        - Renamed 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_tables to 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_tables_700
        - Added 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_tables_800
        - Renamed 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables to 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_700
        - Added 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_800
        - Renamed 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script to 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_700
        - Added 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_800
    - Modified SQL Server DDL Scripts:
        - Renamed 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables to 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables_700
        - Renamed 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables to 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables_700
        - Added 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables_800
        - Renamed 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_tables to 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_tables_700

© Copyright 2000-2017 TIBCO Software Inc.

        - Added 02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_tables_800
        - Renamed 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables to 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_700
        - Added 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_800
        - Renamed 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script to 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_700
        - Added 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script_800

Release 2019Q100 [Jan 7 2019]

- Release Objective: Fixed installation scripts. Added “domainkey” to METRICS_LDAP_PERSON and history tables.
    - Fixed KPImetrics_installation scripts.
        - Modified rebindPhysicalDatabaseType to work with installation scripts.
        - Modified Cache_CPU_MEMORY_CHECKER to put double quotes around windows path.
    - Added domainkey column to qualify a user. The same user may exist in different domains.
        - **** NOTE ****
        - [Perform these actions in order for an existing installation of KPImetrics which contains history data.]
        - [This is not required for new installations.]
            - 1. If the history tables already exist then use alter statements found in /Physical/Physical/KPI_[oracle|sqlserver]/00_fix
                - The alter statements will also modify METRICS_LDAP_PERSON
            - 2. It is required to drop and recreate the staging tables using 02_pqCreateDrop_KPI_Tables_....
            - 3. It is required to drop and recreate the stored procedure using 06_pqCreateDrop_KPI_Plsql_...
            - 4. Update the history tables "domainkey" column with the domain name for all rows.
            - Update the METRICS_LDAP_PERSON table "domainkey" column with the domain name for all rows.
        - Provided the ability to map to multiple LDAP domains
            - Renamed data source /shared/ASAssets/KPImetrics/Physical/Metadata/LDAP to LDAP1
            - Added data source /shared/ASAssets/KPImetrics/Physical/Metadata/LDAP2
        - Modified /Metadata/System/LDAP_PERSON - added "domainkey".
        - Modified /Metadata/System/ClusterSafeCache/Cache_LDAP_PERSON
        - Modified /Physical/Formatting/metrics_resources_usage_hist - added "domainkey".
        - Modified /Business/Logical/userUsage/vLdapPerson - added "domainkey" to the where clause to qualify "user".

© Copyright 2000-2017 TIBCO Software Inc.

- Modified /shared/ASAssets/KPImetrics/Business/Business/requests/ pExceededMemoryPercentRequests to return domain name.
- Modified /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/ pCheckExceedMemoryPercentRequests to qualify userkey with domainkey.
- Modified /shared/ASAssets/KPImetrics/Business/Business/requests/ pLongRunningRequests to return domain name.
- Modified /shared/ASAssets/KPImetrics/Physical/Metadata/System/ClusterSafeCache/ pCheckLongRunningRequests to qualify userkey with domainkey.
- Modified /Metadata/System/ClusterSafeCache/ Cache_AllCustom_AccessByUserOverTime
    - added "domain" to the where clause to qualify "user".
- Modified /Physical/Physical/KPI_oracle/00_fix - alter statements for tables.
- Modified /Physical/Physical/KPI_sqlserver/00_fix - alter statements for tables.
- Modified /Metadata/DDL/Oracle/ 02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables
    - added "domainkey" to metrics_requests_stg, metrics_resources_usage_stg, metrics_sessions_stg
- Modified /Metadata/DDL/Oracle/ 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables
    - added "domainkey" to metrics_requests_hist, metrics_resources_usage_hist, metrics_sessions_hist
- Modified /Metadata/DDL/Oracle/04_pqCreateDrop_KPI_Tables_oracle_kpi_tables
    - added "domainkey" to METRICS_LDAP_PERSON
- Modified /Metadata/DDL/Oracle/ 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script to use "domainkey"
- Modified /Metadata/DDL/SqlServer/02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables
    - added "domainkey" to metrics_requests_stg, metrics_resources_usage_stg, metrics_sessions_stg
- Modified /Metadata/DDL/SqlServer/ 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables
    - added "domainkey" to metrics_requests_hist, metrics_resources_usage_hist, metrics_sessions_hist
- Modified /Metadata/DDL/SqlServer/04_pqCreateDrop_KPI_Tables_sqlserver_kpi_tables
    - added "domainkey" to METRICS_LDAP_PERSON
- Modified /Metadata/DDL/SqlServer/ 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script to use "domainkey"

Release 2018Q401 [Dec 4 2018]

- Release Objective: Deprecated/Removed METRICS_RESOURCES_USAGE_UD.
    - This was a copy of the data from metrics_resources_usage_hist where resourcekind='user defined'. Removed all processing associated with this table and made it a native access.
    - Modified /logical/resourceUsage/vResourceUsageUD to point to metrics_resources_usage_hist.

© Copyright 2000-2017 TIBCO Software Inc.

- Modified /logical/resourceUsage/vResourceUsageAll to select categoryname with no coelesce (bug fix).
- Modified /Physical/KPI_[oracle|sqlserver]/EXCEPT_ACR_AccessByUserOverTime to use metrics_resources_usage_hist.
- Modified /Physical/KPI_[oracle|sqlserver]/EXCEPT_ACR_ActiveResourcesOverPeriodOfTime to use metrics_resources_usage_hist.
- Modified 03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables to add index "mru_hist_reskind_dsname".
- Modified 04_pqCreateDrop_KPI_Tables_oracle_kpi_tables to remove references to METRICS_RESOURCES_USAGE_UD.
- Modified /Oracle_Larger_DataTypes/04_pqCreateDrop_KPI_Tables_oracle_kpi_tables to remove references to METRICS_RESOURCES_USAGE_UD.
- Modified 06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script to remove references to METRICS_RESOURCES_USAGE_UD.
- Modified 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables to add index "mru_hist_reskind_dsname".
- Modified 04_pqCreateDrop_KPI_Tables_sqlserver_kpi_tables to remove references to METRICS_RESOURCES_USAGE_UD.
- Modified 06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script to remove references to METRICS_RESOURCES_USAGE_UD.
- Modified pPurgeData to remove references to METRICS_RESOURCES_USAGE_UD.
- Removed /Formatting/METRICS_RESOURCES_USAGE_UD
- Removed /Physical/Abstraction/METRICS_RESOURCES_USAGE_UD
- Removed /Physical/KPI_oracle/METRICS_RESOURCES_USAGE_UD
- Removed /Metadata/KPI_oracle_11g/CIS_KPI/METRICS_RESOURCES_USAGE_UD
- Removed /Metadata/KPI_oracle_12c/CIS_KPI/METRICS_RESOURCES_USAGE_UD
- Removed /Physical/KPI_sqlserver/METRICS_RESOURCES_USAGE_UD
- Removed /Metadata/KPI_sqlserver_2012/CIS_KPI/dbo/METRICS_RESOURCES_USAGE_UD
- Removed /Metadata/KPI_sqlserver_2014/CIS_KPI/dbo/METRICS_RESOURCES_USAGE_UD

Release 2018Q400 [Dec 1 2018]

- Release Objective: Correlation of published view with datasource lineage.
  - Added processing for SQL request datasource lineage thus allowing reporting on what physical data source resources are related to which published resources.
  - Procedure tuning for Cache_METRICS_SQL_REQUEST_EXEC to block execution when Cache_METRICS_SQL_RESOURCE_LINEAGE is executing so as to not overwhelm the DV node. The opposite is true for Cache_METRICS_SQL_RESOURCE_LINEAGE which checks to see if Cache_METRICS_SQL_REQUEST_EXEC is running before executing.
- <u>Lineage Changes</u>:
- Altered tables METRICS_SQL_RESOURCE and METRICS_SQL_COLUMNS to add START_TIME TIMESTAMP column.
- Altered table METRICS_SQL_RESOURCE to add MESSAGE_TYPE, MESSAGE, PROCESS_NODE_HOST and PROCESSED_NODE_PORT.

© Copyright 2000-2017 TIBCO Software Inc.

- Altered table METRICS_SQL_CONTROL and METRICS_SQL_CONTROL_LOG to add column CONTROL_NAME.
- Added new table METRICS_SQL_RESOURCE_LINEAGE - datasource lineage for each resource found in METRICS_SQL_RESOURCE.
- Added table DDL to 04_pqCreateDrop_KPI_Tables_oracle_kpi_tables and 04_pqCreateDrop_KPI_Tables_sqlserver_kpi_tables.
- Added views and published KPImetrics.requests.vMetricsSqlResourceLineage and vMetricsSqlResourceLineageCountReport.
- Added trigger kpimetricsTrig_23_Cache_METRICS_SQL_RESOURCE_LINEAGE.
- Added cluster safe procedure Cache_METRICS_SQL_RESOURCE_LINEAGE which allows for multi-host processing. This means that node a can process node b's lineage as long as node a has no more rows to process.
- Added variables to commonValues: queryTableLookupDefault, queryTableLookupInterval, getResourceLineagePathExclusionList, numRowsToProcessBeforeExitingLineage
- Modified defaultTriggersToEnable to include new trigger.
- Added index METRICS_SQL_RES_LINEAGE_IDX1 and _IDX2.
- Added index METRICS_SQL_COLUMNS_IDX3.
- Modifed pPurgeData to delete from METRICS_SQL_RESOURCE_LINEAGE.
- General Fixes:
- Fixed a bug in the following view to eliminate ALL_COLUMNS and ALL_PARAMETERS from the result set: /Business/Business/resourceList/vAllPublishedResources.
- Modified pEndWorkflow to better hand error messages with single quotes.
- Fixed updateTriggers which execute the cache when turning off triggers.

Release 2018Q302 [Oct 12 2018]

- Fixed a collection table row volume problem by adding an independent delete trigger [kpimetricsTrig_33_DeleteCollection] for deleting unwanted rows from the collection tables: metrics_requests and metrics_resources_usage. By deleting unwanted rows more often it allows the data transfer script to work more efficiently and quickly.
- Added indexes metrics_resources_usage_idx4, metrics_resources_usage_idx5, metrics_requests_idx1, metrics_requests_idx2 to improve to different styles of delete.
- Modified index METRICS_SQL_REQUEST_IDX4 to try and improve lookup speed.
- Added indexes METRICS_SQL_RESOURCE_IDX2, METRICS_SQL_COLUMNS_IDX2 to try and improve joins.

Release 2018Q301 [Oct 1 2018]

- Modified parseSqlScriptComplex and parseSqlScriptTemplate to change template select from SELECT DISTINCT to SELECT TOP 1 to make it more efficient.

Release 2018Q3 [Sep 2018]

- Modified to allow datasource names in the format of KPI_oracle_11g, KPI_oracle_12c, KPI_sqlserver_2012 or KPI_sqlserver_2014 in order to allow for adding a datasource.
- Modified sqlParserV1_2 to allow for parallel processing of SQL when a cluster is present. A node can process rows other than its own rows when it has no work to do.

© Copyright 2000-2017 TIBCO Software Inc.

- Modified tables:
  - METRICS_SQL_REQUEST - added PROCESSED_NODE_HOST and PROCESSED_NODE_PORT
  - metrics_resources_usage_hist - added resourceorigin
  - METRICS_SQL_RESOURCE - added RESOURCE_ORIGIN
  - METRICS_ALL_RESOURCES - added RESOURCE_ORIGIN
  - METRICS_RESOURCES_USAGE_UD - added RESOURCE_ORIGIN
- Added tables:
  - metrics_sessions_stg - staging table for metrics_sessions_hist
  - metrics_requests_stg - staging table for metrics_requests_hist
  - metrics_resources_usage_stg - staging table for metrics_resources_usage_hist
  - METRICS_SQL_CONTROL - used to control parallel processing for sqlParserV1_2
  - METRICS_SQL_CONTROL_LOG - logs the activity if turned if debugging is turned on.
- Modified sqlParserV1_2:
  - Allow for parallel processing of SQL when a cluster is present. A node can process rows other than its own rows when it has no work to do.
  - Added ability to process web service calls for operations that are TABLES or PROCEDURES.
- Modified /shared/ASAssets/KPImetrics/Business/Logical/metrics *_hist views to use where starttime > TO_TIMESTAMP('1900-01-01 00:00:00.000') to force the use of parallel queries.
- Consolidated P_METRICS_ALL_TABLES_exec from KPI_oracle/KPI_sqlserver to /ClusterSafeCache/pMETRICS_ALL_TABLES_exec.
- Consolidated P_PARTITION_MANAGER_exec from KPI_oracle/KPI_sqlserver to /ClusterSafeCache/pPARTITION_MANAGER_exec.
- Consolidated the 3 DBMS triggers each from KPI_oracle/KPI_sqlserver to a single set under ClusterSafeTriggers.
- Removed GetSequenceValueProc from KPI_oracle/KPI_sqlserver and consolidated the functionality into /Abstraction/GetSequenceValueProc.
- Moved the "update METRICS_RESOURCES_USAGE_UD where RESOURCE_ID IS NULL" logic from Cache_ALL_RESOURCES to the data transfer script.
- Removed the "update metrics_resources_usage_hist where RESOURCE_ID IS NULL" logic from Cache_ALL_RESOURCES alltogher to avoid updates on the history table.
- Consolidated 05_pqCreateDrop_KPI_Tables_[oracle|sqlserver]_kpi_application_tables into 04_pqCreateDrop_KPI_Tables_[oracle|sqlserver]_kpi_tables
- Renamed 06_pqCreateDrop_KPI_Tables_[oracle|sqlserver]_kpi_sequence to 05_pqCreateDrop_KPI_Tables_[oracle|sqlserver]_kpi_sequence
- Renamed 07_pqCreateDrop_KPI_Plsql_[oracle|sqlserver]_data_xfer_script to 06_pqCreateDrop_KPI_Plsql_[oracle|sqlserver]_data_xfer_script
- Modified 06_pqCreateDrop_KPI_Plsql_[oracle|sqlserver]_data_xfer_script: [inserts are faster than deletes]
  - Modified the Oracle and SQL Server data transfer scripts to use staging tables for insert/update/deletes in order to only perform inserts into history tables.
  - Eliminated the delete job filter SQL on metrics_resources_usage by including it in the insert stage where clause.

© Copyright 2000-2017 TIBCO Software Inc.

  - Eliminated the delete not exist SQL on metrics_requests by including a WHERE EXISTS in the insert stage where clause.

- Renamed 08_pqInsert_KPI_Tables_METRICS_JOB_tables to 07_pqInsert_KPI_Tables_METRICS_JOB_tables
- Renamed 09_pqInsert_KPI_Tables_METRICS_EVENT_REGISTRATION to 08_pqInsert_KPI_Tables_METRICS_EVENT_REGISTRATION
- Modified commonValues:
  - Removed dataSourceSchemaPath_oracle, dataSourceSchemaPath_sqlserver.
  - Removed cisCaseSensitivity as it is no longer needed.
  - Added dataSourceCatalog, dataSourceSchema, historyTableCompression.
  - Added dataSourceCollation_sqlserver which gets applied at the time of creating the SQL Server tables to allow for case sensitive searches and joins on columns.

© Copyright 2000-2017 TIBCO Software Inc.

9 Appendix A – Partitioning Schemes

This section provides information on the various database partition schemes.

Oracle Partition Scheme

This section describes how Oracle partitioning is utilized. Oracle by far provides the most elegant and easy to implement solution for partitioning. There are very few moving parts and the commands are all inclusive. The following description demonstrates the "Oracle Partition Management Sliding Window Scenario". It shows the SQL statements that get executed for creation, adding and dropping of partitions.

Step 1. Configure commonValues

Setup of the \Configuration\commonValues is a key aspect for partitioning. The partitionNumber and partitionStartDate define how many partitions will be initially created and managed and when the first partition start date is configured for. In the example below there will be 3 partitions created starting with 20170501.

commonValues.partitionNumber=3

commonValues.partitionStartDate='2017-05-01'

Therefore, the history tables will be created as follows:

Metrics History Table	Partition Name	Partition Rule
metrics_requests_hist	MR201705	< 2017-06-01 00:00:00
	MR201706	< 2017-07-01 00:00:00
	MR201707	< 2017-08-01 00:00:00
metrics_resources_usage_hist	MRU201705	< 2017-06-01 00:00:00
	MRU201706	< 2017-07-01 00:00:00
	MRU201707	< 2017-08-01 00:00:00
metrics_sessions_hist	MS201705	< 2017-06-01 00:00:00
	MS201706	< 2017-07-01 00:00:00
	MS201707	< 2017-08-01 00:00:00

Step 2. Create initial history tables, partition strategy and indexes:

[03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables]

Create the history table for the respective tables. For Oracle, the create table statement contains the syntax for creating the partitions. Notice that the PARTITION BY RANGE is used on the "starttime" column. Since the partitionNumber=3 then 3 partitions are created for the initial partition starting with partitionStartDate=2017-05-01. Each partition has a unique name which describes what bucket of data it contains. The partition also contains a rule to compare the data to determine which bucket it goes in.

```
CREATE TABLE "CIS_KPI"."metrics_requests_hist" (<column_list>)
```

© Copyright 2000-2017 TIBCO Software Inc.

LOB ("description") STORE AS (TABLESPACE "METRICS_DATA_HIST" ENABLE STORAGE IN ROW CHUNK 8192 PCTVERSION 10 NOCACHE NOLOGGING)

LOB ("message") STORE AS (TABLESPACE "METRICS_DATA_HIST" ENABLE STORAGE IN ROW CHUNK 8192 PCTVERSION 10 NOCACHE NOLOGGING)

NOCOMPRESS TABLESPACE "METRICS_DATA_HIST" RESULT_CACHE (MODE DEFAULT) PCTUSED 0 PCTFREE 10 INITRANS 1 MAXTRANS 255

STORAGE (BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)

PARTITION BY RANGE ("starttime")

(

PARTITION MR201705 VALUES LESS THAN (TIMESTAMP' 2017-06-01 00:00:00')

LOGGING NOCOMPRESS TABLESPACE "METRICS_DATA_HIST"

LOB ("description") STORE AS (TABLESPACE "METRICS_DATA_HIST" ENABLE STORAGE IN ROW CHUNK 8192 RETENTION NOCACHE LOGGING

STORAGE (INITIAL 8M NEXT 1M MINEXTENTS 1 MAXEXTENTS UNLIMITED PCTINCREASE 0 BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT))

LOB ("message") STORE AS (TABLESPACE "METRICS_DATA_HIST" ENABLE STORAGE IN ROW CHUNK 8192 RETENTION NOCACHE LOGGING

STORAGE (INITIAL 8M NEXT 1M MINEXTENTS 1 MAXEXTENTS UNLIMITED PCTINCREASE 0 BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT))

PCTFREE 10 INITRANS 1 MAXTRANS 255

STORAGE (INITIAL 8M NEXT 1M MAXSIZE UNLIMITED MINEXTENTS 1 MAXEXTENTS UNLIMITED BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT),

PARTITION MR201706 VALUES LESS THAN (TIMESTAMP' 2017-07-01 00:00:00')

LOGGING NOCOMPRESS TABLESPACE "METRICS_DATA_HIST"

LOB ("description") STORE AS (TABLESPACE "METRICS_DATA_HIST" ENABLE STORAGE IN ROW CHUNK 8192 RETENTION NOCACHE LOGGING

STORAGE (INITIAL 8M NEXT 1M MINEXTENTS 1 MAXEXTENTS UNLIMITED PCTINCREASE 0 BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT))

LOB ("message") STORE AS (TABLESPACE "METRICS_DATA_HIST" ENABLE STORAGE IN ROW CHUNK 8192 RETENTION NOCACHE LOGGING

STORAGE (INITIAL 8M NEXT 1M MINEXTENTS 1 MAXEXTENTS UNLIMITED PCTINCREASE 0 BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT))

PCTFREE 10 INITRANS 1 MAXTRANS 255

STORAGE (INITIAL 8M NEXT 1M MAXSIZE UNLIMITED MINEXTENTS 1 MAXEXTENTS UNLIMITED BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT),

PARTITION MR201707 VALUES LESS THAN (TIMESTAMP' 2017-08-01 00:00:00')

LOGGING NOCOMPRESS TABLESPACE "METRICS_DATA_HIST"

© Copyright 2000-2017 TIBCO Software Inc.

```
  LOB ("description") STORE AS (TABLESPACE "METRICS_DATA_HIST"   ENABLE STORAGE IN ROW
CHUNK 8192   RETENTION   NOCACHE   LOGGING
      STORAGE (INITIAL 8M   NEXT 1M   MINEXTENTS 1   MAXEXTENTS UNLIMITED   PCTINCREASE 0
BUFFER_POOL DEFAULT   FLASH_CACHE DEFAULT   CELL_FLASH_CACHE DEFAULT))
  LOB ("message") STORE AS (TABLESPACE "METRICS_DATA_HIST"   ENABLE STORAGE IN ROW
CHUNK 8192   RETENTION   NOCACHE   LOGGING
      STORAGE (INITIAL 8M   NEXT 1M   MINEXTENTS 1   MAXEXTENTS UNLIMITED   PCTINCREASE 0
BUFFER_POOL DEFAULT   FLASH_CACHE DEFAULT   CELL_FLASH_CACHE DEFAULT))
  PCTFREE 10   INITRANS 1   MAXTRANS 255
  STORAGE (INITIAL 8M   NEXT 1M   MAXSIZE UNLIMITED   MINEXTENTS 1   MAXEXTENTS
UNLIMITED   BUFFER_POOL DEFAULT   FLASH_CACHE DEFAULT   CELL_FLASH_CACHE DEFAULT)
)
NOCACHE NOPARALLEL MONITORING;
```

Create the indexes for the history tables. Indexes are created/managed on the main table. Notice that the tablespace can be different for indexes than for the history tables.

```
CREATE INDEX "mr_hist_rid" ON "CIS_KPI"."metrics_requests_hist" ("requestid", "nodehost", "nodeport")
TABLESPACE "METRICS_DATA_IDX";
CREATE INDEX "mr_hist_rid_time" ON "CIS_KPI"."metrics_requests_hist" ("requestid", "starttime",
"nodehost", "nodeport") TABLESPACE "METRICS_DATA_IDX";
```

Repeat all creation operations shown above for “metrics_requests_usage_hist” and “metrics_sessions_hist”.

This shows the partition distribution for the history tables.

table_name	partition_number	num_rows
"MR201705"	1	0
"MR201706"	2	0
"MR201707"	3	0
"MRU201705"	1	0
"MRU201706"	2	0
"MRU201707"	3	0
"MS201705"	1	0
"MS201706"	2	0
"MS201707"	3	0

Step 3. Add next month partition: [/KPI_oracle/pqPartition_metrics_history_tables_ADD]

Notice how a new partition table is created simply with an ALTER TABLE statement. No other operation is required. Indexes are automatically created.

```
ALTER TABLE "CIS_KPI"."metrics_requests_hist" ADD
  PARTITION "MR201708" VALUES LESS THAN (TIMESTAMP '2017-09-01 00:00:00')
  LOGGING NOCOMPRESS TABLESPACE “METRICS_DATA_COLL”
```

© Copyright 2000-2017 TIBCO Software Inc.

LOB ("description") STORE AS (TABLESPACE “METRICS_DATA_COLL” ENABLE STORAGE IN ROW CHUNK 8192 RETENTION NOCACHE LOGGING

STORAGE (INITIAL 8M NEXT 1M MINEXTENTS 1 MAXEXTENTS UNLIMITED PCTINCREASE 0 BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT))

LOB ("message") STORE AS (TABLESPACE “METRICS_DATA_COLL” ENABLE STORAGE IN ROW CHUNK 8192 RETENTION NOCACHE LOGGING

STORAGE (INITIAL 8M NEXT 1M MINEXTENTS 1 MAXEXTENTS UNLIMITED PCTINCREASE 0 BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT))

PCTFREE 10 INITRANS 1 MAXTRANS 255

STORAGE (INITIAL 8M NEXT 1M MAXSIZE UNLIMITED MINEXTENTS 1 MAXEXTENTS UNLIMITED BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT);

Repeat all creation operations shown above for “metrics_requests_usage_hist” and “metrics_sessions_hist”.

This shows the partition distribution for the history tables. Notice how 201708 partitions were added.

table_name	partition_number	num_rows
"MR201705"	1	0
"MR201706"	2	0
"MR201707"	3	0
"MR201708"	4	0
"MRU201705"	1	0
"MRU201706"	2	0
"MRU201707"	3	0
"MRU201708"	4	0
"MS201705"	1	0
"MS201706"	2	0
"MS201707"	3	0
"MS201708"	4	0

Step 4. Insert test rows:
[/shared/ASAssets/KPImetrics/Physical/Metadata/DML/test_insert_metrics_requests_hist]

Notice how there is 1 row in each metrics_requests_hist partitions.

© Copyright 2000-2017 TIBCO Software Inc.

table_name	partition_number	num_rows
"MR201705"	1	1
"MR201706"	2	1
"MR201707"	3	1
"MR201708"	4	1
"MRU201705"	1	0
"MRU201706"	2	0
"MRU201707"	3	0
"MRU201708"	4	0
"MS201705"	1	0
"MS201706"	2	0
"MS201707"	3	0
"MS201708"	4	0

Step 5. Drop the oldest partition.

Test the Drop Scenario by forcing the number of partitions to be 1 less than before with the date incremented by 1 month

commonValues.partitionNumber=2

commonValues.partitionStartDate='2017-06-01'

Drop the oldest partition: [/KPI_oracle/pqPartition_metrics_history_tables_DROP]

The strategy for dropping a partition is called the sliding window. It is much more efficient than deleting rows. KPImetrics simply drops the partition. It is very simple.

```
ALTER TABLE "CIS_KPI"."metrics_requests_hist" DROP PARTITION "MR201705";
```

Repeat the same steps for "metrics_resources_usage_hist" and "metrics_sessions_hist":

As shown below, the number of partitions has been reduced from 4 to 3. The Jun 1 2017 boundary representing May 2017 data has been dropped. The oldest partition was dropped. The remaining data shown below is still properly partitioned as expected.

cluster	nodehost	nodeport	requestid	parentid	sessionid	requesttype	description	starttime	endtime	totaldurati...
[NULL]	lsdnecvbe...	9500	2	1	1	JDBC	[NULL]	2017-06-01 07:00...	2017-06-...	0
[NULL]	lsdnecvbe...	9500	3	1	1	JDBC	[NULL]	2017-07-01 07:00...	2017-07-...	0
[NULL]	lsdnecvbe...	9500	4	1	1	JDBC	[NULL]	2017-08-01 07:00...	2017-08-...	0

table_name	partition_number	num_rows
"MR201706"	1	1
"MR201707"	2	1
"MR201708"	3	1
"MRU201706"	1	0
"MRU201707"	2	0
"MRU201708"	3	0
"MS201706"	1	0
"MS201707"	2	0
"MS201708"	3	0

© Copyright 2000-2017 TIBCO Software Inc.

Step 6. Repeat Drop oldest partition.

Now, repeat the drop exercise one more time

commonValues.partitionNumber=1

commonValues.partitionStartDate='2017-07-01'

Drop the oldest partition: [/KPI_oracle/pqPartition_metrics_history_tables_DROP]

```
ALTER TABLE "CIS_KPI"."metrics_requests_hist" DROP PARTITION "MR201706";
```

As shown below, the number of partitions has been reduced from 3 to 2. The Jul 1 2017 boundary representing June 2017 data has been dropped. The table representing the oldest partition was dropped. The remaining data shown below is still properly partitioned as expected.

cluster	nodehost	nodeport	requestid	parentid	sessionid	requesttype	description	starttime	endtime	totaldurati...
[NULL]	lsdnecvbe...	9500	3	1	1	JDBC	[NULL]	2017-07-01 07:00:...	2017-07-...	0
[NULL]	lsdnecvbe...	9500	4	1	1	JDBC	[NULL]	2017-08-01 07:00:...	2017-08-...	0

table_name	partition_number	num_rows
"MR201707"	1	1
"MR201708"	2	1
"MRU201707"	1	0
"MRU201708"	2	0
"MS201707"	1	0
"MS201708"	2	0

© Copyright 2000-2017 TIBCO Software Inc.

SQL Server Partition Scheme

This section describes how SQL Server partitioning is utilized. The scenario that is explained is here is referred to as the "SQL Server Partition Management Sliding Window Scenario". It shows the SQL statements that get executed for creation, adding and dropping of partitions.

Step 1. Configure commonValues

Setup of the \Configuration\commonValues is a key aspect for partitioning. The partitionNumber and partitionStartDate define how many partitions will be initially created and managed and when the first partition start date is configured for. In the example below there will be 3 partitions created starting with 20170501.

commonValues.partitionNumber=3

commonValues.partitionStartDate='2017-05-01'

Therefore the history tables will be created as follows:

Metrics History Table	Derived Partition Name	Partition Function Rule
metrics_requests_hist	MR201705	< 2017-06-01 00:00:00
	MR201706	< 2017-07-01 00:00:00
	MR201707	< 2017-08-01 00:00:00
metrics_resources_usage_hist	MRU201705	< 2017-06-01 00:00:00
	MRU201706	< 2017-07-01 00:00:00
	MRU201707	< 2017-08-01 00:00:00
metrics_sessions_hist	MS201705	< 2017-06-01 00:00:00
	MS201706	< 2017-07-01 00:00:00
	MS201707	< 2017-08-01 00:00:00

Note that in SQL Server there is no such thing as a partition name. It is simply shown here as the "Derived Partition Name" to describe how the partitions of data are distributed. It is shown fore reporting purposes in the procedure 03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables_ROW_DISTRIBUTION.

Step 2. Create initial history tables, partition strategy and indexes:

[03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables]

Create the partition function and partition scheme. This partition strategy uses "RANGE RIGHT" as it makes it easier to define a monthly bucket based on the 1st of the month at midnight. All data less than that value is placed in the bucket for the current month. Also notice that the partitioning scheme applies the filegroup. This provides the flexibility of placing each month partition in its own filegroup and potentially disk spindle. However, for the sake of ease of implementation, KPImetrics only implements a single file group for all partitions. These days, the underlying disk architecture is usually hidden anyway with NFS mounted drives.

```
CREATE PARTITION FUNCTION "mr_hist_partition_function" (DATETIME2(3)) AS RANGE RIGHT FOR
VALUES ('2017-06-01 00:00:00', '2017-07-01 00:00:00', '2017-08-01 00:00:00');
```

© Copyright 2000-2017 TIBCO Software Inc.

```
CREATE PARTITION SCHEME "mr_hist_partition_scheme" AS PARTITION "mr_hist_partition_function" ALL
TO ([METRICS_DATA_HIST]);
```

Create the partitioned history table, partitioned archive table and partitioned indexes for the respective tables. The archive table must be created exactly like the history table in order for the SWITCH to take place. This includes the partition scheme and indexes. Notice how the tables are created based on the partition scheme and not the filegroup. The filegroup is actually assigned to the partition scheme as shown previously.

```
CREATE TABLE "dbo"."metrics_requests_hist_arch" (<column_list>) ON
"mr_hist_partition_scheme"("starttime");

CREATE TABLE "dbo"."metrics_requests_hist" (<column_list>) ON "mr_hist_partition_scheme"("starttime");
```

Create partitioned indexes on history table. The indexes are created on the same partitioning scheme as the tables.

```
IF NOT EXISTS(SELECT * FROM sys.indexes WHERE object_id = object_id(N'metrics_requests_hist') AND
NAME ='mr_hist_rid') CREATE NONCLUSTERED INDEX "mr_hist_rid" ON "dbo"."metrics_requests_hist"
("requestid", "nodehost", "nodeport") ON "mr_hist_partition_scheme"("starttime");

IF NOT EXISTS(SELECT * FROM sys.indexes WHERE object_id = object_id(N'metrics_requests_hist') AND
NAME ='mr_hist_rid_time') CREATE NONCLUSTERED INDEX "mr_hist_rid_time" ON
"dbo"."metrics_requests_hist" ("requestid", "starttime", "nodehost", "nodeport") ON
"mr_hist_partition_scheme"("starttime");
```

Create partitioned indexes on archive table. The archive table must look exactly like the history table to perform SWITCH.

```
IF NOT EXISTS(SELECT * FROM sys.indexes WHERE object_id = object_id(N'metrics_requests_hist_arch')
AND NAME ='mr_hist_rid_arch') CREATE NONCLUSTERED INDEX "mr_hist_rid_arch" ON
"dbo"."metrics_requests_hist_arch" ("requestid", "nodehost", "nodeport") ON
"mr_hist_partition_scheme"("starttime");

IF NOT EXISTS(SELECT * FROM sys.indexes WHERE object_id = object_id(N'metrics_requests_hist_arch')
AND NAME ='mr_hist_rid_time_arch') CREATE NONCLUSTERED INDEX "mr_hist_rid_time_arch" ON
"dbo"."metrics_requests_hist_arch" ("requestid", "starttime", "nodehost", "nodeport") ON
"mr_hist_partition_scheme"("starttime");
```

Repeat table and index creation for “metrics_requests_usage_hist” and “metrics_sessions_hist”.

This shows the partition distribution for the history tables

© Copyright 2000-2017 TIBCO Software Inc.

table_name	boundary_definition	partition_number	num_rows
metrics_requests_hist	Jun 1 2017 12:00AM	1	0
metrics_requests_hist	Jul 1 2017 12:00AM	2	0
metrics_requests_hist	Aug 1 2017 12:00AM	3	0
metrics_resources_usage_hist	Jun 1 2017 12:00AM	1	0
metrics_resources_usage_hist	Jul 1 2017 12:00AM	2	0
metrics_resources_usage_hist	Aug 1 2017 12:00AM	3	0
metrics_sessions_hist	Jun 1 2017 12:00AM	1	0
metrics_sessions_hist	Jul 1 2017 12:00AM	2	0
metrics_sessions_hist	Aug 1 2017 12:00AM	3	0

Step 3. Add next month partition: [/KPI_sqlserver/pqPartition_metrics_history_tables_ADD]

Notice how the SPLIT RANGE is used to add the next month range. If the current month is 2017-07 [Jul] then the boundary is 2017-08. The next month is 20170-08 [Aug] and the boundary is 2017-09 [Sep]. The following statement alters the scheme with NEXT USED [filegroup] to force the new partition range to take affect.

```
ALTER PARTITION FUNCTION "mr_hist_partition_function"() SPLIT RANGE ('2017-09-01 00:00:00');

ALTER PARTITION SCHEME "mr_hist_partition_scheme" NEXT USED [METRICS_DATA_HIST];

ALTER PARTITION FUNCTION "mru_hist_partition_function"() SPLIT RANGE ('2017-09-01 00:00:00');

ALTER PARTITION SCHEME "mru_hist_partition_scheme" NEXT USED [METRICS_DATA_HIST];

ALTER PARTITION FUNCTION "ms_hist_partition_function"() SPLIT RANGE ('2017-09-01 00:00:00');

ALTER PARTITION SCHEME "ms_hist_partition_scheme" NEXT USED [METRICS_DATA_HIST];
```

Step 4. Insert test rows:
[/shared/ASAssets/KPImetrics/Physical/Metadata/DML/test_insert_metrics_requests_hist]

table_name	boundary_definition	partition_number	num_rows
metrics_requests_hist	Jun 1 2017 12:00AM	1	1
metrics_requests_hist	Jul 1 2017 12:00AM	2	1
metrics_requests_hist	Aug 1 2017 12:00AM	3	1
metrics_requests_hist	Sep 1 2017 12:00AM	4	1
metrics_resources_usage_hist	Jun 1 2017 12:00AM	1	0
metrics_resources_usage_hist	Jul 1 2017 12:00AM	2	0
metrics_resources_usage_hist	Aug 1 2017 12:00AM	3	0
metrics_resources_usage_hist	Sep 1 2017 12:00AM	4	0
metrics_sessions_hist	Jun 1 2017 12:00AM	1	0
metrics_sessions_hist	Jul 1 2017 12:00AM	2	0
metrics_sessions_hist	Aug 1 2017 12:00AM	3	0
metrics_sessions_hist	Sep 1 2017 12:00AM	4	0

Notice that there is 1 row in each partition

Result rows:1 - 4

cluster	nodehost	nodeport	requestid	parentid	sessionid	requesttype	description	starttime	endtime	totaldurati...
[NULL]	lsdnecvbe...	9500	1	1	1	JDBC	[NULL]	2017-05-22 07:57:21.407	2017-05-...	0
[NULL]	lsdnecvbe...	9500	1	1	1	JDBC	[NULL]	2017-06-22 07:57:21.407	2017-06-...	0
[NULL]	lsdnecvbe...	9500	2	1	1	JDBC	[NULL]	2017-07-22 07:57:21.407	2017-07-...	0
[NULL]	lsdnecvbe...	9500	3	1	1	JDBC	[NULL]	2017-08-22 07:57:21.407	2017-08-...	0

© Copyright 2000-2017 TIBCO Software Inc.

Step 5. Drop the oldest partition.

Test the Drop Scenario by forcing the number of partitions to be 1 less than before with the date incremented by 1 month

commonValues.partitionNumber=2

commonValues.partitionStartDate='2017-06-01'

Drop the oldest partition: [/KPI_sqlserver/pqPartition_metrics_history_tables_DROP]

The strategy for dropping a partition is called the sliding window. It is much more efficient than deleting rows. The idea is to SWITCH the oldest partition with an empty archive table which is configured with the same structure and indexes. This is accomplished with the ALTER TABLE SWITCH PARTITION shown below.

```
ALTER TABLE "dbo"."metrics_requests_hist" SWITCH PARTITION 1 TO "dbo"."metrics_requests_hist_arch"
PARTITION 1;
```

The next part of the strategy is to perform the ALTER FUNCTION MERGE RANGE to merge the empty partition with the next oldest partition. This will be very fast since the oldest partition designated by the boundary range “2017-06-01 00:00:00” is empty.

```
ALTER PARTITION FUNCTION mr_hist_partition_function() MERGE RANGE ('2017-06-01 00:00:00');
```

Finally, the archive table which now contains the rows from the oldest partition of the history table is truncated to make it empty. Truncating is very fast compared to deleting rows. Theoretically, if it is required, the rows could be archived off to off-line storage or a Big Data solution like Hadoop.

```
TRUNCATE TABLE "dbo"."metrics_requests_hist_arch";
```

Repeat the same steps for "metrics_resources_usage_hist" and "metrics_sessions_hist":

```
ALTER TABLE "dbo"."metrics_resources_usage_hist" SWITCH PARTITION 1 TO
"dbo"."metrics_resources_usage_hist_arch"  PARTITION 1;

ALTER PARTITION FUNCTION mru_hist_partition_function() MERGE RANGE ('2017-06-01 00:00:00');

TRUNCATE TABLE "dbo"."metrics_resources_usage_hist_arch";

ALTER TABLE "dbo"."metrics_sessions_hist" SWITCH PARTITION 1 TO "dbo"."metrics_sessions_hist_arch"
PARTITION 1;

ALTER PARTITION FUNCTION ms_hist_partition_function() MERGE RANGE ('2017-06-01 00:00:00');

TRUNCATE TABLE "dbo"."metrics_sessions_hist_arch";
```

© Copyright 2000-2017 TIBCO Software Inc.

As shown below, the number of partitions has been reduced from 4 to 3. The Jun 1 2017 boundary representing May 2017 data has been dropped. The data in the partition was switched with an empty partitioned archive table. The data in the archive table was truncated.

The remaining data shown below is still properly partitioned as expected.

cluster	nodehost	nodeport	requestid	parentid	sessionid	requesttype	description	starttime	endtime	totaldurati...
[NULL]	lsdnecvbe...	9500	1	1	1	JDBC	[NULL]	2017-06-22 07:57:21.407	2017-06-...	0
[NULL]	lsdnecvbe...	9500	2	1	1	JDBC	[NULL]	2017-07-22 07:57:21.407	2017-07-...	0
[NULL]	lsdnecvbe...	9500	3	1	1	JDBC	[NULL]	2017-08-22 07:57:21.407	2017-08-...	0

table_name	boundary_definition	partition_number	num_rows
metrics_requests_hist	Jul 1 2017 12:00AM	1	1
metrics_requests_hist	Aug 1 2017 12:00AM	2	1
metrics_requests_hist	Sep 1 2017 12:00AM	3	1
metrics_resources_usage_hist	Jul 1 2017 12:00AM	1	0
metrics_resources_usage_hist	Aug 1 2017 12:00AM	2	0
metrics_resources_usage_hist	Sep 1 2017 12:00AM	3	0
metrics_sessions_hist	Jul 1 2017 12:00AM	1	0
metrics_sessions_hist	Aug 1 2017 12:00AM	2	0
metrics_sessions_hist	Sep 1 2017 12:00AM	3	0

Step 6. Repeat Drop oldest partition.

Now, repeat the drop exercise one more time

commonValues.partitionNumber=1

commonValues.partitionStartDate='2017-07-01'

Drop the oldest partition: [/KPI_sqlserver/pqPartition_metrics_history_tables_DROP]

```
ALTER TABLE "dbo"."metrics_requests_hist" SWITCH PARTITION 1 TO "dbo"."metrics_requests_hist_arch"
PARTITION 1;

ALTER PARTITION FUNCTION mr_hist_partition_function() MERGE RANGE ('2017-07-01 00:00:00');

TRUNCATE TABLE "dbo"."metrics_requests_hist_arch";

ALTER TABLE "dbo"."metrics_resources_usage_hist" SWITCH PARTITION 1 TO
"dbo"."metrics_resources_usage_hist_arch"  PARTITION 1;

ALTER PARTITION FUNCTION mru_hist_partition_function() MERGE RANGE ('2017-07-01 00:00:00');

TRUNCATE TABLE "dbo"."metrics_resources_usage_hist_arch";

ALTER TABLE "dbo"."metrics_sessions_hist" SWITCH PARTITION 1 TO "dbo"."metrics_sessions_hist_arch"
PARTITION 1;

ALTER PARTITION FUNCTION ms_hist_partition_function() MERGE RANGE ('2017-07-01 00:00:00');

TRUNCATE TABLE "dbo"."metrics_sessions_hist_arch";
```

© Copyright 2000-2017 TIBCO Software Inc.

As shown below, the number of partitions has been reduced from 3 to 2. The Jul 1 2017 boundary representing June 2017 data has been dropped. The data in the partition was switched with an empty partitioned archive table. The data in the archive table was truncated.

The remaining data shown below is still properly partitioned as expected.

cluster	nodehost	nodeport	requestid	parentid	sessionid	requesttype	description	starttime	endtime	totaldurati...
[NULL]	lsdnecvbe...	9500	2	1	1	JDBC	[NULL]	2017-07-22 07:57:21.407	2017-07-...	0
[NULL]	lsdnecvbe...	9500	3	1	1	JDBC	[NULL]	2017-08-22 07:57:21.407	2017-08-...	0

table_name	boundary_definition	partition_number	num_rows
metrics_requests_hist	Aug 1 2017 12:00AM	1	1
metrics_requests_hist	Sep 1 2017 12:00AM	2	1
metrics_resources_usage_hist	Aug 1 2017 12:00AM	1	0
metrics_resources_usage_hist	Sep 1 2017 12:00AM	2	0
metrics_sessions_hist	Aug 1 2017 12:00AM	1	0
metrics_sessions_hist	Sep 1 2017 12:00AM	2	0

© Copyright 2000-2017 TIBCO Software Inc.

10 Post-Installation Background Information

Information Only Section

This section provides a background on what gets executed by the post-installation script.

Script Requirements:
1. This script must be executed by the user admin or someone who has admin privileges
2. The KPImetrics CAR file has been imported either manually or with 1_Pre_Installation.

Steps to be performed after the KPImetrics CAR file is imported:
1. Update impacted resources.
2. Set resource privileges
3. Enable and test the KPImetrics data source
4. Rebind physical database type tables to the data source schema
5. Rebind the physical abstraction folder "/Physical/Abstraction" to the correct physical data source folder "/Physical/KPI_[oracle|sqlserver]"
6. Remove CIS_KPI or DVKPI folder if not needed
7. Update impacted resources
8. Validate and Create KPImetrics collection tables exist
9. Create KPImetrics cache tables, sequences and procedures [This will drop existing]
10. Introspect / Reintrospect the KPImetrics data source
11. Change resource ownership
12. Refresh the LDAP_PERSON view
13. Load the METRICS_JOBS table
14. Load the METRICS_EVENT_REGISTRATION table
15. Install the CpuAndMemCheckerCjp scripts into the file system for either Windows or UNIX.
16. Update the CPUAndMemChecker data source url and reintrospect it
17. Test the /System/CPU_MEMORY_CHECKER view
18. Installation and configuration are complete

How the scripts work:

The KPImetrics module requires several tables in the KPImetrics data source database in order to store metrics data for reporting. You must create these storage tables using the provided DDL in order for the KPImetrics module to function correctly.

If you choose to create the tables from within Studio, execute the procedures under /shared/ASAssets/KPImetrics/Physical/Metadata/DDL for your data source type. Proceed to the following section associated for your database type.

© Copyright 2000-2017 TIBCO Software Inc.

Procedure Parameters:

IN displayDDL – Y=Display the DDL, N=Do not display the DDL.

IN executeDDL – Y=execute the DDL, N=Do not execute the DDL. If you choose to execute the DDL externally, you will need to execute each of the 01-07 DDL procedures with the variable set as executeDDL=N and displayDDL=Y. This will output the DDL that you need to execute externally without actually performing the DDL operations. Once you have executed each procedure 01-07, you are now ready to run the DDL externally. Once you have executed the DDL proceed to the next section "*Common Configuration for all Databases*".

IN dropIndexes– Y=drop the indexes before creating the first. N=do not drop the indexes.

IN dropTables – Y=drop the tables before creating the tables. N=do not drop the tables.

IN createTables – Y=execute the table creation DDL, N=display the table creation DDL in the console window only.

IN createIndexes– Y=execute index creation DDL. N=display the index creation DDL in the console window only.

OUT cursCombinedResult – Provides a status on each SQL statement executed.

OUT sqlScript – Generates an output of the entire script which can be used for external execution.

The common 08-09 DML procedures to populate the database must be run from within DV as there is no external SQL generation for those.

Create the KPImetrics storage tables for Oracle

1. Do the DV metrics collection tables exist?
    a. NO
        i. Create a tablespace called "METRICS_DATA_COLL" with NOLOGGING.
            1. Example:
            2. create tablespace METRICS_DATA_COLL nologging datafile 'C:/DV/oracle/metrics_data_coll01.dbf' size 500m autoextend on next 50m extent management local;
        ii. Execute ***02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_stage_tables_[700|800]***

© Copyright 2000-2017 TIBCO Software Inc.

1. SchemaName – derived from commonValues.dataSourceSchema
2. TablespaceName – derived from commonValues.collectionTablespaceName

iii. Execute ***02_pqCreateDrop_KPI_Tables_oracle_metrics_collection_tables_[700|800]***

1. SchemaName – derived from commonValues.dataSourceSchema
2. TablespaceName – derived from commonValues.collectionTablespaceName

b. YES

i. Consider exporting the data and turning off metrics in order to create these tables with NO LOGGING option enabled in their own tablespace called “METRICS_DATA_COLL”.

ii. If this is not desired, then bypass this step.

2. Create the metrics history tables and indexes

a. Create a tablespace called “METRICS_DATA_HIST” and “METRICS_DATA_IDX” with logging turned on.

i. Example:

ii. create tablespace METRICS_DATA_HIST logging datafile 'C:/DV/oracle/metrics_data_hist01.dbf' size 500m autoextend on next 50m extent management local;

iii. create tablespace METRICS_DATA_IDX logging datafile 'C:/DV/oracle/metrics_data_idx01.dbf' size 500m autoextend on next 50m extent management local;

b. Drop and Create the metrics history tables and indexes: metrics_requests_hist, metrics_resources_usage_hist and metrics_sessions

c. Execute ***03_pqCreateDrop_KPI_Tables_oracle_metrics_history_tables_[700|800]*** with input:

i. partitionNumber– derived from commonValues.partitionNumber

ii. partitionStartDate– derived from commonValues.partitionStartDate

iii. SchemaName – derived from commonValues.dataSourceSchema

iv. TablespaceName – derived from commonValues.historyTablespaceName and commonValues.indexTablespaceName.

© Copyright 2000-2017 TIBCO Software Inc.

3. Create the metrics KPI reporting tables and indexes
    a. Execute ***04_pqCreateDrop_KPI_Tables_oracle_kpi_tables*** with input:
        i. SchemaName – derived from commonValues.dataSourceSchema
        ii. TablespaceName – derived from commonValues.historyTablespaceName and commonValues.indexTablespaceName.
4. Create the metrics KPI reporting sequence
    a. Execute ***05_pqCreateDrop_KPI_Tables_oracle_kpi_sequence*** with input:
        i. SchemaName – derived from commonValues.dataSourceSchema
5. Create the metrics KPI "native" PLSQL data transfer script
    a. Execute ***06_pqCreateDrop_KPI_Plsql_oracle_data_xfer_script_[700|800]*** with no input:
        i. userTransformation – Contains a SQL-based transformation that is valid within the context of SELECT and uses "user" as the column name to transform. It must be a valid transformation for Oracle. If null, the default is simply "user". The column name that is "user" is transformed into the "userkey". The column "userkey" is joined with METRICS_LDAP_PERSON.
        ii. SchemaName – derived from commonValues.dataSourceSchema

Create the KPImetrics storage tables for SQL Serer

1. Do the DV metrics collection tables exist?
    a. NO
        i. Create a tablespace (filegroup) called "METRICS_DATA_COLL".
            1. <u>Example</u>:
            2. alter database CIS_KPI add filegroup METRICS_DATA_COLL;
            3. alter database CIS_KPI add FILE (NAME='METRICS_DATA_COLL1', FILENAME='C:\SQLServer_FileGroup\METRICS_DATA_COLL1') TO FILEGROUP METRICS_DATA_COLL;
        ii. Execute ***02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_stage_tables_[700|800]***
            1. SchemaName – derived from commonValues.dataSourceSchema
            2. TablespaceName – derived from commonValues.collectionTablespaceName

© Copyright 2000-2017 TIBCO Software Inc.

iii. Execute ***02_pqCreateDrop_KPI_Tables_sqlserver_metrics_collection_tables_[700|800]***
    1. SchemaName – derived from commonValues.dataSourceSchema
    2. TablespaceName – derived from commonValues.collectionTablespaceName

b. YES
    i. Consider exporting the data and turning off metrics in order to create these tables with NO LOGGING option enabled in their own tablespace called "METRICS_DATA_COLL".
    ii. If this is not desired, then bypass this step.

2. Create the metrics history tables and indexes
    a. Create a tablespace (filegroup) called "METRICS_DATA_HIST" and "METRICS_DATA_IDX" with logging turned.
        i. Example:
        ii. alter database CIS_KPI add filegroup METRICS_DATA_HIST;
        iii. alter database CIS_KPI add FILE (NAME='METRICS_DATA_HIST_1', FILENAME='C:\SQLServer_FileGroup\METRICS_DATA_HIST_1') TO FILEGROUP METRICS_DATA_HIST;
        iv. alter database CIS_KPI add filegroup METRICS_DATA_IDX;
        v. alter database CIS_KPI add FILE (NAME='METRICS_DATA_IDX_1', FILENAME='C:\SQLServer_FileGroup\METRICS_DATA_IDX_1') TO FILEGROUP METRICS_DATA_IDX;
    b. Drop and Create the metrics history tables and indexes: metrics_requests_hist, metrics_resources_usage_hist and metrics_sessions
    c. Execute ***03_pqCreateDrop_KPI_Tables_sqlserver_metrics_history_tables*** with input:
        i. partitionNumber– derived from commonValues.partitionNumber
        ii. partitionStartDate– derived from commonValues.partitionStartDate
        iii. SchemaName – derived from commonValues.dataSourceSchema
        iv. TablespaceName – derived from commonValues.historyTablespaceName and commonValues.indexTablespaceName.
3. Create the metrics KPI reporting tables and indexes
    a. Execute ***04_pqCreateDrop_KPI_Tables_sqlserver_kpi_tables*** with input:

© Copyright 2000-2017 TIBCO Software Inc.

i. SchemaName – derived from commonValues.dataSourceSchema
ii. TablespaceName – derived from commonValues.historyTablespaceName and commonValues.indexTablespaceName.

4. Create the metrics KPI reporting sequence
   a. Execute ***05_pqCreateDrop_KPI_Tables_sqlserver_kpi_sequence*** with input:
      i. SchemaName – derived from commonValues.dataSourceSchema
5. Create the metrics KPI “native” PLSQL data transfer script
   a. Execute ***06_pqCreateDrop_KPI_Plsql_sqlserver_data_xfer_script*** with no input:
      i. userTransformation – Contains a SQL-based transformation that is valid within the context of SELECT and uses "user" as the column name to transform. It must be a valid transformation for SQL Server. If null, the default is simply "user". The column name that is "user" is transformed into the "userkey". The column "userkey" is joined with METRICS_LDAP_PERSON.
      ii. SchemaName – derived from commonValues.dataSourceSchema

Common Configuration for all Databases

1. Execute the procedure /shared/ASAssets/KPImetrics/Configuration/**updateEnableDatasource**
   a. This procedure will enable the proper KPImetrics data source [KPI_oracle or KPI_sqlserver] based on commonValues configuration.
2. Reintrospect or Introspect the data source
   a. Was the default CIS_KPI schema or CIS_KPI/dbo catalog/schema used?
      i. YES: Reintrospect the KPImetrics data source to confirm that the tables are visible.
         1. Right-click on your chosen data source “KPI_<database_type>” and select “Re-Introspect Now” and wait for it to complete.
         2. Click OK when completed.
      ii. NO: A new schema or catalog name was used. Introspect the new schema or catalog/schema.
         1. Right-click on your chosen data source “KPI_<database_type>” and select “Add/Remove Tables”.
         2. Select your schema or catalog/schema

© Copyright 2000-2017 TIBCO Software Inc.

3. Select all tables/procedures:
    a. P_METRICS_ALL_TABLES
    b. “cache_status” and “cache_tracking”
    c. Starting with “METRICS_...”
    d. Starting with “metrics_...”
    e. Click Next. Click Finish. Wait for the introspection to complete. Review the list of tables in this section:
    f. “Metadata Data Source for KPI_<database_type>”
4. Click OK when completed.

3. Execute the rebind of the physical database type folder resources /shared/ASAssets/KPImetrics/Configuration/**rebindPhysicalDatabaseType**
    a. This procedure rebinds the /shared/ASAssets/KPImetrics/Physical/Physical/KPI_<database_type> folder to the configured schema and catalog path found in commonValues.
    b. No input is required. It uses commonValues to determine the target data source to rebind to. *Note*: /shared/ASAssets/KPImetrics/Configuration/commonValues must be configured prior to executing this procedure.

4. Execute the rebind procedure /shared/ASAssets/KPImetrics/Configuration/**rebindPhysicalAbstraction** to rebind all KPImetrics abstraction layer views to the appropriate data source.
    a. No input is required. It uses commonValues to determine the target data source to rebind to. *Note*: /shared/ASAssets/KPImetrics/Configuration/commonValues must be configured prior to executing this procedure.
    b. Based on the commonValues.dataSourceName, it will rebind the /Physical/Abstraction views to either /Physical/KPI_oracle or /Physical/KPI_sqlserver.
    c. Additionally, there is a list of views “commonValues.viewOptimizationPathList” used by rebindPhysicalAbstraction, that is used to modify the view SQL Script to optimize for Oracle or SQL Server in order to achieve push-down of the query.

5. Remove default CIS_KPI schema/catalog if not used.
    a. If a different schema/catalog was chosen other than CIS_KPI then remove the old

© Copyright 2000-2017 TIBCO Software Inc.

CIS_KPI name from your chosen data source “KPI_<database_type>”.

i. Right-click on CIS_KPI and select delete

1. Update Impacted Resources

b. Potentially, there may be some resources that are red/impacted.

ii. Session is null

iii. Java.lang.null

6. Execute /shared/ASAssets/KPImetrics/Configuration/**updateImpacteResources**
   a. Refresh your studio once this completes and the red/impacted resources should disappear. If they do not, then edit the ones that are still red/impacted. Put a space anywhere in the resource and save the resource. The act of editing and changing the resource should cause the common error “session is null” to go away. If the error persists, perhaps it is some other issue that requires a closer look.

7. Execute ***07_pqInsert_KPI_Tables_METRICS_JOB_tables*** with no input
   a. Note: This same procedure is used to modify rows. It always deletes the rows and then inserts the rows.

8. Execute ***08_pqInsert_KPI_Tables_METRICS_EVENT_REGISTRATION*** with input:
   a. deleteAllRows – Y=delete all rows first, N=Do not delete all rows. (default).
   b. This same procedure is used to modify rows. First delete the rows and then insert the rows.

© Copyright 2000-2017 TIBCO Software Inc.