Enterprise Planning Batch Facility

Technical Design Document

Date Created: May 4, 2020

Date Updated: May 4, 2020

**Version:** 2020.1.0.0



**Legal Notice**

Copyright © 2020, Blue Yonder, Inc. All rights reserved. Blue Yonder is a Registered Trademark of Blue Yonder, Inc. All other company and product names may be Trademarks, Registered Trademarks or Service Marks of the companies with which they   
are associated. Blue Yonder reserves the right at any time and without notice to change these materials or any of the functions, features or specifications of any of the software described herein. Blue Yonder shall have no warranty obligation with respect   
to these materials or the software described herein, except as approved in Blue Yonder’s Software License Agreement with an authorized licensee.

Table of Contents

[Document Control 1](#_Toc39674339)

[The EP Batch Facility Section 1 2](#_Toc39674340)

[What is the EP Batch Facility? 2](#_Toc39674341)

[EP Batch Facility Architecture Illustrated 2](#_Toc39674342)

[What is the Adhoc Batch Facility? 3](#_Toc39674343)

[Adhoc Batch Facility Architecture Illustrated 3](#_Toc39674344)

[The EP Batch Environment Section 2 4](#_Toc39674345)

[Batch Command Console Applications 4](#_Toc39674346)

[EKBF Server Commands 4](#_Toc39674347)

[Batch Scripts 4](#_Toc39674348)

[EP Batch Service 5](#_Toc39674349)

[EP Batch Client 5](#_Toc39674350)

[IBM DataStage 5](#_Toc39674351)

[Mulesoft 5](#_Toc39674352)

[Batch Scheduler 5](#_Toc39674353)

[Batch Command Console Applications Section 3 6](#_Toc39674354)

[Batch Initialize (obsolete) 9](#_Toc39674355)

[Batch Get Current Task (obsolete) 10](#_Toc39674356)

[Batch Restart Status (obsolete) 10](#_Toc39674357)

[Batch Set Current Task (obsolete) 11](#_Toc39674358)

[Copy Fact Data 11](#_Toc39674359)

[Sample FDE XML specification 11](#_Toc39674360)

[Create SFTP Control Files (Cloud) 13](#_Toc39674361)

[Sample SFTPControlTemplate.ctl file 13](#_Toc39674362)

[Current EKB Time 13](#_Toc39674363)

[Dump Errors 15](#_Toc39674364)

[Get Input Source Name 15](#_Toc39674365)

[Foundation Restart Status 16](#_Toc39674366)

[Latest Fact Data Time 16](#_Toc39674367)

[Process Adhoc Batch (obsolete) 17](#_Toc39674368)

[Quartz Restart Status 17](#_Toc39674369)

[Quartz Sync Status 18](#_Toc39674370)

[Row Count 18](#_Toc39674371)

[Run SQL 19](#_Toc39674372)

[Schedule Adhoc Batch (obsolete) 19](#_Toc39674373)

[Send REST Request (Mulesoft) 20](#_Toc39674374)

[Start Process 20](#_Toc39674375)

[Truncate JI Errors (Mulesoft & Datastage) 23](#_Toc39674376)

[Encrypted Password Repository 24](#_Toc39674377)

[EKBF Server Commands Section 4 25](#_Toc39674378)

[Foundation Data Remove (FDR) 25](#_Toc39674379)

[Foundation Data Update (FDU) 25](#_Toc39674380)

[FDU Transaction History Utility 25](#_Toc39674381)

[Foundation Set Time (FST) 25](#_Toc39674382)

[Foundation Structure Update (FSU) 26](#_Toc39674383)

[Quartz Data Remove (QDR) 26](#_Toc39674384)

[Quartz Sync Foundation (Quartz Structure Sync/Quartz Data Sync) 26](#_Toc39674385)

[EP Mid-Tier Admin Client Console 27](#_Toc39674386)

[Batch Scripts Section 5 28](#_Toc39674387)

[Actuals Cube Remove Data 28](#_Toc39674388)

[Actuals Cube Sync 29](#_Toc39674389)

[Actuals Data Update 30](#_Toc39674390)

[Allow Client Connections 32](#_Toc39674391)

[Archive Logs 32](#_Toc39674392)

[Clear Import Tables (Mulesoft) 33](#_Toc39674393)

[Create AG Tables (Mulesoft) 34](#_Toc39674394)

[Cube Purge Aged Data 35](#_Toc39674395)

[Cube Remove Data 36](#_Toc39674396)

[Cube Sync 37](#_Toc39674397)

[EKB Cleanup 38](#_Toc39674398)

[FDE Data Update 38](#_Toc39674399)

[FDE Stage Fact Data 39](#_Toc39674400)

[Foundation Actuals Data Update 40](#_Toc39674401)

[Foundation Plan Data Update 41](#_Toc39674402)

[Foundation Purge Aged Data 42](#_Toc39674403)

[Foundation Set Time 43](#_Toc39674404)

[Foundation Structure Update 44](#_Toc39674405)

[Gather Schema Stats 45](#_Toc39674406)

[Sample GatherSchemaStats.sql 45](#_Toc39674407)

[Get Batch Result (Mulesoft) 46](#_Toc39674408)

[Get EKB Schema Config (Mulesoft) 47](#_Toc39674409)

[Get EXT Schema Config (Mulesoft) 48](#_Toc39674410)

[Global Variables 48](#_Toc39674411)

[Init EXT Schema (Mulesoft) 50](#_Toc39674412)

[Initialize SFTP (Cloud) 51](#_Toc39674413)

[JI Member Data Load (Mulesoft & DataStage) 52](#_Toc39674414)

[JI Meta Data Load (Mulesoft & Datastage) 53](#_Toc39674415)

[Pending Batch Processes (Mulesoft) 55](#_Toc39674416)

[Plan Cube Sync 56](#_Toc39674417)

[Plan Data Update 56](#_Toc39674418)

[Process Member Data (Mulesoft) 58](#_Toc39674419)

[Process Meta Data (Mulesoft) 59](#_Toc39674420)

[Quartz Cube Sync 60](#_Toc39674421)

[Quartz Purge Aged Data 61](#_Toc39674422)

[Rebuild Contexts 62](#_Toc39674423)

[Set JI Mule Properties (Mulesoft) 63](#_Toc39674424)

[Sample SetEKBSchemaProperties.json 63](#_Toc39674425)

[Sample SetEmailProperties.json 64](#_Toc39674426)

[Sample SetEXTSchemaProperties.json 64](#_Toc39674427)

[Sample SetFrameworkProperties.json 64](#_Toc39674428)

[Sample SetHTTPProperties.json 65](#_Toc39674429)

[Sample SetJIProperties.json 65](#_Toc39674430)

[Stage Facts (Cloud) 66](#_Toc39674431)

[Stage Structure (Cloud) 66](#_Toc39674432)

[Start EP Services 67](#_Toc39674433)

[Stop EP Services 68](#_Toc39674434)

[EP Batch Service Section 6 69](#_Toc39674435)

[EP Batch Client/Server Workflow 70](#_Toc39674436)

[Suggested Batch Service Settings 71](#_Toc39674437)

[Sample Application Settings 72](#_Toc39674438)

[EP Batch Client Section 7 74](#_Toc39674439)

[EP Batch Client Desktop 74](#_Toc39674440)

[Menu Items 74](#_Toc39674441)

[EP Batch Client Session 75](#_Toc39674442)

[Options Panel 75](#_Toc39674443)

[Sample Application Settings 76](#_Toc39674444)

[Sample User Settings 76](#_Toc39674445)

[EP Batch Facility Installation Section 8 78](#_Toc39674446)

[EP Batch Client and EP Batch Service 78](#_Toc39674447)

[Batch Command Console Applications and Scripts 78](#_Toc39674448)

# Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Updated By | Change Applied |
| 1.0 | 2/22/2013 | John Acuna | Initial Document Creation. |
| 1.1 | 3/26/2013 | John Acuna | Review comments from Andrew Kenney. Addition of CopyFactData, QuartzSyncStatus, BatchInitialize, BatchGetCurrentTask, BatchRestartStatus and BatchSetCurrentTask. |
| 1.2 | 3/31/2013 | John Acuna | Added CurrentEKBTime. |
| 1.3 | 6/24/2013 | John Acuna | Added CubeRemoveData.bat and ActualsCubeRemoveData.bat |
| 1.4 | 7/19/2013 | John Acuna | Added AllowClientConnections.bat |
| 1.5 | 10/14/2013 | John Acuna | Added allowed error codes to RowCount command and GlobalVariables. |
| 1.6 | 11/15/2013 | John Acuna | Added ArchiveLogs expiration logic to purge old log archives. |
| 1.7 | 12/10/2013 | John Acuna | Added EKBCleanup and GetInputSourceName. |
| 1.8 | 1/16/2014 | John Acuna | Added the Adhoc Batch Facility |
| 1.9 | 3/26/2014 | John Acuna | Added Foundation and Quartz purge aged data functionality. |
| 1.10 | 6/25/2014 | John Acuna | Added LatestFactDataTime and modified FoundationSetTime to add option of setting Foundation time to match latet set of loaded fact data. |
| 1.11 | 3/20/2018 | John Acuna | Added DumpErrors.exe, CreateSFTPControlFiles and InitializeSFTP.bat. |
| 1.12 | 7/26/2018 | John Acuna | Enhancements to DumpErrors.exe, StartProcesss.exe, ActualsDataUpdate and PlanDataUpdate. |
| 2018.4.0.0 | 10/9/2018 | John Acuna | Added RunSQL.exe. |
| 2020.1.0.0 | 5/4/2020 | John Acuna | Added support for JI on Mulesoft |

# The EP Batch Facility Section 1

## What is the EP Batch Facility?

At the core of EP is the Enterprise Knowledge Base (EKB). EKB is the repository for both Hierarchical and Fact data. Hierarchy data is stored in the EKB Structure (EKBS) schema while Fact data (both Actual and User Plan submitted) is stored in the EKB Foundation (EKBF) schema[[1]](#footnote-2). Additionally, aggregated fact data is calculated and stored in EKB Quartz (EKBQ) schemas for Posted Versions only.

EKBS, EKBF and EKBQ data must be maintained on a regular basis through an automated mechanism called the EP Batch Facility. The EP Batch Facility is comprised of elements of the EP product and third party products as well as custom built utilities and batch scripts.

The overall Batch Workflow includes the sourcing of data into input sources and the eventual processing of that data. This section describes the mechanism that processes the staged data.

## EP Batch Facility Architecture Illustrated



## What is the Adhoc Batch Facility?

Some implementations require EP batch jobs to be scheduled on an irregular basis at the desire of the EP user. The Adhoc Batch Facility provides a mechanism to queue batch jobs on the fly that are intended to be processed as part of the nightly or weekend batch run.

The typical usage of the Adhoc Batch Facility is expected to include the queueing of jobs via an EP Client automation script and processing of the job queue using the customer’s batch scheduler (e.g. Control-M).

## Adhoc Batch Facility Architecture Illustrated



# The EP Batch Environment Section 2

## Batch Command Console Applications

In support of batch scripts (.bat files), the EP Batch Facility provides several command line utiltities. These utilities are:

* **BatchInitialize (obsolete) –** Creates a new Batch Workflow or re-initializes an existing Batch Workflow.
* **BatchGetCurrentTask (obsolete) –** Gets the current task of a Batch Workflow.
* **BatchRestartStatus (obsolete) –** Determines if a Batch Workflow can be started or requires a restart/recovery.
* **BatchSetCurrentTask (obsolete) –** Sets the current task of a Batch Workflow.
* **CopyFactData** – Extracts data from a SubDomain and loads the results into an input source.
* **CreateSFTPControlFiles (Cloud)** – Creates actuals and structure tables SQLLDR control files.
* **CurrentEKBTime** – Determines the current Foundation Calendar Member or Calendar Member from an offset of the current Member.
* **DumpErrors** – Dumps the contents of the actuals or structure error tables and any failed records in staging. Can also be used to dump the obsolete retained feature member names as well as the contents of all tables and views that match a wildcard string name.
* **GetInputSourceName** – Gets the input source name of a Fact Table Family (FTF).
* **FoundationRestartStatus** – Determines if an EKBF server command can be started or requires a restart/recovery.
* **LatestFactDataTime –** Either returns the Time Member ID of the most recent data in a Fact Table Family (Granite) Table or determines whether fact data had been loaded within the last number of hours specified.
* **ProcessAdhocBatch (obsolete)** – Processes the scheduled Jobs associated with a Job Group.
* **QuartzRestartStatus** – Determines if an EKBQ server command can be started or requires a restart/recovery.
* **QuartzSyncStatus** – Determines if a Quartz Cube needs to be structure and/or data synchronized.
* **RowCount** – Returns the number of rows in a database table, useful in determining if errors records were written.
* **RunSQL** – Runs an SQL statement or PL/SQL against a known datasource without having to provide clear text passwords. If there is a result set, a header is returned followed by data rows delimited with commas.
* **ScheduleAdhocBatch (obsolete)** - Adds a Job to a Job Group, essentially queueing it for future execution by the ProcessAdhocBatch command.
* **Send REST Request (Mulesoft)** – Sends GET, POST, PUT and DELETE web requests with parameters to an endpoint.
* **StartProcess** – Spawns a collection of command lines in background processes and monitors them until completion.
* **TruncateJIErrors (Mulesoft & Datastage)** – Clears the JI errors table of all rows in preparation for a fresh JI job run.

Most of these utilities require either Foundation or Quartz schema passwords. Because the EP Batch Facility is not officially part of the EP Product offering, access to the encrypted password repository within EP is not allowed. Therefore, the EP Batch Facility maintains its own encrypted password repository described later in this document.

## EKBF Server Commands

Server Commands perform a variety of EKBF and EKBQ maintenance tasks which include the following:

* Propagation of EKBS changes into EKBF
* Processing of staged Fact data into Granite
* Aggregation of Granite data into Quartz
* Setting of Foundation time
* etc.

The EP Batch Facility only supports the Windows implementation of the EP Servers Commands.

## Batch Scripts

Batch Scripts are utilized to combine the functionality of Console Applications, Server Commands and other Batch Scripts to accomplish a process workflow.

## EP Batch Service

The EP Batch Service is a Windows Service that runs on every EP Mid-Tier. It provides the capability of running EP Batch Facility Scripts from a remote system through a TCP connection. This negates the need for a remote desktop into a Mid-Tier system and isolates a user from the inner workings and configuration of the EP Batch Facility.

The EP Batch Service also ensures that only one batch script is running on a Mid-Tier system at a time. This is important to avoid process failures due to Server Command lock contention.

## EP Batch Client

The EP Batch Client is a Windows GUI Application that is used to send requests to and view the response from the EP Batch Service. The EP Batch Client can be installed on any system that has TCP access to the EP Mid-Tier system running the EP Batch Service.

## IBM DataStage

DataStage is a third-party Extract, Transform, Load (ETL) product used to process staged data from the EXT (a.k.a. JI) Schema into EKBS. The EP product provides scripts packaged into a DataStage project called the JDA Integrator (JI). These scripts utilize the EKBS Batch Import Facility to process staged data into EKBS Member and Meta Data tables.

The EP Batch Facility runs two JI scripts:

* **seqDBMASTER\_EXT\_MetaData\_TO\_EPSD\_MetaData** – Processes staged data in the EXT\_\* tables into EKBS.
* **seqDBMASTER\_EXT\_TO\_EPServerData** – Processes staged data in the AG\_\* tables into EKBS.

IBM DataStage has been replaced by Mulesoft in EP 2019.1.

## Mulesoft

Mulesoft is a third-party software integration development environment for the creation of web service based APIs. The purpose of these APIs is data integration between applications.

Blueyonder APIs, built upon Mulesoft, process staged data from the EXT (a.k.a. JI) Schema into EKBS. These API’s utilize the EKBS Batch Import Facility to process staged data into EKBS Member and Meta Data tables.

Mulesoft replaced IBM DataStage as of EP 2019.1

## Batch Scheduler

A Scheduler provides the capability to run a Batch Workflow on a timed interval. In a typical EP implementation, there are nightly and weekend Workflows to provide the regular maintenance of the EKB Database.

JDA does not provide or endorse any particular Batch Scheduler vendor or product. The selection of a Scheduler Product is entirely up to the JDA Customer.

# Batch Command Console Applications Section 3

The Batch Command Console Applications are written against the .NET 4.0 Framework and require the downloadable installation found at…

<http://www.microsoft.com/en-us/download/details.aspx?id=17718>

A subset of console applications is related to the implementation of Workflow restart capability. The intent is to provide wrapper scripts (i.e. Daily, Saturday, Sunday, etc.) the ability to leave “breadcrumbs” as they execute and to be able to efficiently recover from failure scenarios. The following console applications are related to Workflow restart:

* Batch Initialize
* Batch Get Current Task
* Batch RestartStatus
* Batch Set Current Task

The EP Batch Facility provides the framework for script recovery, but usage of the framework is up to the wrapper script itself.

**Sample Wrapper Script Template**:

@ECHO OFF

REM #

REM # Created by John Acuna Version 1.0.0.0 on 3/26/2013

REM #

REM # Revision History

REM #

REM # Version Author Comments

REM #

REM # ==============================================================================

REM # Purpose:

REM # Executes the Daily Batch Workflow

REM # ==============================================================================

REM #

REM # BatchRestartStatus ERROR CODES

REM #

REM # ARGUMENT\_MISSING = -9002

REM # CANNOT\_DETERMINE = -9001

REM # NO\_RESTART\_PENDING = 9000

REM # RESTART\_PENDING = 9001

REM # PROCESS\_RUNNING = 9002

REM #

REM # BatchInitialize ERROR CODES

REM #

REM # ARGUMENT\_MISSING = -9002

REM # CANNOT\_INITIALIZE = -9001

REM # BATCH\_INITIALIZED = 9000

REM # ==============================================================================

SET WORKFLOW=Daily

CALL GlobalVariables.bat

ECHO The %WORKFLOW% batch workflow has started

BatchRestartStatus %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW%

IF %ERRORLEVEL% EQU 9000 (

SET ERRORLEVEL=0

BatchInitialize %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW% FSU FDU QS

IF ERRORLEVEL 9001 (

ECHO The %WORKFLOW% batch workflow has failed to initialize

GOTO :EXIT

)

GOTO :FSU

)

IF %ERRORLEVEL% EQU 9001 (

BatchGetCurrentTask %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW%

IF ERRORLEVEL 3 (

GOTO :QS

)

IF ERRORLEVEL 2 (

GOTO :FDU

)

IF ERRORLEVEL 1 (

GOTO :FSU

)

ECHO The %WORKFLOW% batch workflow task id %ERRORLEVEL% is unknown on restart

GOTO :EXIT

)

ECHO The %WORKFLOW% batch workflow has failed trying to determine restart status

GOTO :EXIT

:FSU

BatchSetCurrentTask %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW% FSU

CALL FoundationStructureUpdate.bat

IF %ERRORLEVEL% NEQ 0 (

ECHO The %WORKFLOW% batch workflow has failed performing the FSU task

GOTO :EXIT

)

:FDU

BatchSetCurrentTask %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW% FDU

CALL FoundationPlanDataUpdate.bat

IF %ERRORLEVEL% NEQ 0 (

ECHO The %WORKFLOW% batch workflow has failed performing the FDU task

GOTO :EXIT

)

BatchSetCurrentTask %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW%

:QS

BatchSetCurrentTask %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW% QS

CALL PlanCubeSync.bat

IF %ERRORLEVEL% NEQ 0 (

ECHO The %WORKFLOW% batch workflow has failed performing the QS task

GOTO :EXIT

)

BatchSetCurrentTask %FOUNDATIONUSER% %DATASOURCE% %WORKFLOW%

ECHO The %WORKFLOW% batch workflow has succeeded

:EXIT

EXIT %ERRORLEVEL%

## Batch Initialize (obsolete)

**Mid-Tier**: Primary and Secondary

**Purpose**: To Create a new Batch Workflow or re-initialize an existing Batch Workflow. Each task will be added with a Status of NOT\_STARTED. If the BATCH\_PROCTRACK and BATCH\_TASKTRACK tables do not exist, they will be created.

**Usage**: BatchInitialize [user] [data source] [workflow name] [task name]…

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| workflow name | A unique identifier for the Batch Workflow |
| task name | One or more unique identifiers of the tasks the Workflow is comprised of. They are specified in the order in which they will be run within the Workflow. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [workflow name] [task name]… |
| -9001 | Cannot initialize batch |
| 9000 | Batch initialized |

**BATCH\_PROCTRACK Table**:

PROC\_ID NUMBER NOT NULL,

HOST\_NAME VARCHAR2(255 BYTE) NOT NULL,

USER\_NAME VARCHAR2(255 BYTE) NOT NULL,

WORKFLOW\_NAME VARCHAR2(30 BYTE) NOT NULL,

CURRENT\_TASK\_NAME VARCHAR2(30 BYTE)

**BATCH\_TASKTRACK Table**:

TASK\_ID NUMBER NOT NULL,

TASK\_NAME VARCHAR2(30 BYTE) NOT NULL,

WORKFLOW\_NAME VARCHAR2(30 BYTE) NOT NULL,

STATUS VARCHAR2(30 BYTE) DEFAULT 'NOT\_STARTED' NOT NULL,

START\_TIME DATE,

END\_TIME DATE

## Batch Get Current Task (obsolete)

**Mid-Tier**: Primary and Secondary

**Purpose**: To Get the current task of a Batch Workflow.

**Usage**: BatchGetCurrentTask [user] [data source] [workflow name]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| workflow name | The name of an existing Batch Workflow |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [workflow name] |
| -9001 | Cannot get current task for Workflow |
| 0 | No Current Task for Workflow |
| > 0 | The Task Id of the Workflow’s current Task. The Task Id is a sequence assigned to the initialized Workflow Task Names. |

## Batch Restart Status (obsolete)

**Mid-Tier**: Primary and Secondary

**Purpose**: To determine if a Batch Workflow can be started or requires a restart/recovery.

**Usage**: BatchRestartStatus [user] [data source] [workflow name]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| workflow name | The name of an existing Batch Workflow |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [workflow name] |
| -9001 | Cannot determine if restart is pending |
| 9000 | Restart not pending for Workflow |
| 9001 | Restart pending for Workflow |
| 9002 | Workflow is currently running |

## Batch Set Current Task (obsolete)

**Mid-Tier**: Primary and Secondary

**Purpose**: To set the current task of a Batch Workflow. Sets the task Status to IN\_PROGRESS and sets the Start Time to now. Changes any previous current task to a Status of COMPLETED and sets the End Time to now. Setting the current task to null indicates the completion of the Workflow.

**Usage**: BatchSetCurrentTask [user] [data source] [workflow name] [task name]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| workflow name | The name of an existing Batch Workflow |
| task name | The name of an existing Batch Workflow Task |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [workflow name] [task name] |
| -9001 | Cannot set current task |
| 9000 | Current task set |

## Copy Fact Data

**Mid-Tier**: Primary only

**Purpose**: To extract fact data from a Subdomain using the Foundation Data Export (FDE) and load the results into a target input source. Fields are automatically mapped from the export result set to the input source.

**Usage**: CopyFactData [user] [data source] [fde xml path] [input source] [period increment] [filter sql path]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| fde xml path | Pathname to a FDE export specification that defines what data will be extracted from a Subdomain |
| Input source | The target input source that extracted data will be written to |
| period increment | A positive or negative offset used to derive a time period relative to the current Foundation time |
| filter sql path | Pathname to a SQL statement that will be executed after fact data has been copied. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [fde xml path] [input source] [period increment] |
| -9001 | Cannot copy data |
| 9000 | Records copied successfully |
| 9001 | FDE did not return any data |
| 9002 | No matching measures between FDE returned data and target input source |

#### Sample FDE XML specification

The FDE requires an XML specification that describes what measure values are to be extracted from a Subdomain. The FDE is described in detail within the EP Server Administrator Guide.

Below is a sample of an XML specification that pulls measure values from three FTFs within the same Subdomain and accumulates the measure values into a single set of measures that would be used to load the target input source by the CopyFactData command.

<?xml version="1.0"?>

<PKBExportDefinition>

<SubdomainName>PLAN\_FDE\_STCT</SubdomainName>

<GlobalIDFormat>EUCLIDS</GlobalIDFormat>

<DimLevels>

<DimLevel>

<Dimension>CALENDAR</Dimension>

<OutputMemberIDFormat>TECHKEYS</OutputMemberIDFormat>

<ScopeLevel>WEEK</ScopeLevel>

<ScopeSelections>

<MemberSelection>

<MemberId>[current week member]</MemberId>

<MemberCount>[period increment]</MemberCount>

</MemberSelection>

</ScopeSelections>

</DimLevel>

<DimLevel>

<Dimension>ORGANIZATION</Dimension>

<OutputMemberIDFormat>TECHKEYS</OutputMemberIDFormat>

<ScopeLevel>CHANNEL</ScopeLevel>

</DimLevel>

<DimLevel>

<Dimension>PRODUCT</Dimension>

<OutputMemberIDFormat>TECHKEYS</OutputMemberIDFormat>

<ScopeLevel>CLASS</ScopeLevel>

</DimLevel>

</DimLevels>

<Measures>

<Measure>

<MeasureName>ISCC</MeasureName>

<MeasureExpn>ISCC\_STCA+ISCC\_STMX+ISCC\_STUS</MeasureExpn>

</Measure>

<Measure>

<MeasureName>ISC</MeasureName>

<MeasureExpn>ISC\_STCA+ISC\_STMX+ISC\_STUS</MeasureExpn>

</Measure>

<Measure>

<MeasureName>SCD</MeasureName>

<MeasureExpn>SCD\_STCA+SCD\_STMX+SCD\_STUS</MeasureExpn>

</Measure>

<Measure>

<MeasureName>SSCC</MeasureName>

<MeasureExpn>SSCC\_STCA+SSCC\_STMX+SSCC\_STUS</MeasureExpn>

</Measure>

<Measure>

<MeasureName>SSC</MeasureName>

<MeasureExpn>SSC\_STCA+SSC\_STMX+SSC\_STUS</MeasureExpn>

</Measure>

</Measures>

</PKBExportDefinition>

Note that the above example demonstrates the usage of two placeholders:

* **[current week member]** – Will be substituted with the current Foundation Week Member Id
* **[period increment]** – Will be substituted with the “period increment” argument value if supplied. The default value is 0.

## Create SFTP Control Files (Cloud)

**Mid-Tier**: Primary only

**Purpose**: Creates SQL Loader control files used when processing Structure Member and Fact data files from the SFTP Inbox.

**Usage**: CreateSFTPControlFiles [user] [data source] [output path] [structure/actuals]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| output path | The directory that the control files will be written to. |
| structure/actuals | ‘structure’ will create control files for the JI staging tables and ‘actuals’ will create control files for all actuals staging tables. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [output path] [structure/actuals] |
| -9001 | Dump failed |
| 9000 | No records found in any {0} tables. |
| 9001 | Number of {0} table dump files written: {1} |

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [output path] [structure/actuals] |
| -9001 | Create SQLLDR control file failed |
| 9000 | No eligible {0} tables found to create SQLLDR control files for |
| 9001 | Number of {0} table SQLLDR control files written: {1} |

#### Sample SFTPControlTemplate.ctl file

OPTIONS (skip=1)

LOAD DATA

INFILE 'placeholder.dat'

BADFILE 'placeholder.bad'

TRUNCATE

INTO TABLE {0}

FIELDS TERMINATED BY '|' OPTIONALLY ENCLOSED BY '"'

TRAILING NULLCOLS

(

{1}

)

## Current EKB Time

**Mid-Tier**: Primary only

**Purpose**: To determine the current Foundation Calendar Member or Calendar Member from an offset of the current Member.

**Usage**: CurrentEKBTime [user] [data source] [level]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| level | The Level to retrieve the current Calendar Member from. Using ‘show\_all’ will return all current Calendar Members. Using ‘week\_of\_month’ will return the week number of the current week (system clock) within the fiscal calendar month. Using an offset will return a Calendar Member relative to the current Member. For example, week+1 or month-12. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [level] |
| -9001 | Cannot determine current time |

**Sample Output:**

CurrentEKBTime ekbf epdev show\_all

Member              Level               Start Date

------------------- ------------------- ----------

TYR\_PRE             Ttl\_Yrs               2/3/2008

F2012\_AR\_U3         Ttl\_AE\_AYr           6/10/2012

Y2012               Year                 1/29/2012

S2012\_01            Season               1/29/2012

Y2012\_AE            AltYr\_AE            12/25/2011

C2012\_AE\_SMR        Collect\_AE            4/1/2012

Q2012\_02            Quarter              4/29/2012

TAYR\_AE\_PRE         Collect\_AR          12/26/2010

Y2012\_AR            Tot\_CA\_IL           12/25/2011

M2012\_05            Month                5/27/2012

F2012\_AE\_U2         FlrSet\_AE            5/20/2012

C2012\_AR\_SMR        FlrSet\_AR            3/25/2012

W2012\_20            Week                 6/10/2012

TAYR\_AR\_PRE         CA\_IL               12/26/2010

D2012\_134           Day                  6/10/2012

CurrentEKBTime ekbf epdev week

W2012\_20

CurrentEKBTime ekbf epdev week+1

W2012\_21

CurrentEKBTime ekbf epdev week-1

W2012\_19

CurrentEKBTime ekbf epdev month

M2012\_05

CurrentEKBTime ekbf epdev month-24

M2010\_05

CurrentEKBTime ekbf epdev flrset\_ae+2

F2012\_AE\_BU

CurrentEKBTime ekbf epdev week\_of\_month

3

## Dump Errors

**Mid-Tier**: Primary only

**Purpose**: Dumps the contents of the actuals or structure error tables and any failed records in staging. Can also be used to dump the obsolete retained feature member names as well as the contents of all tables and views that match a wildcard string name.

**Usage**: DumpErrors [user] [data source] [output path] [structure/actuals/obsolete/wildcard] [wildcard table name]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| output path | The directory that the dump files will be written to. |
| structure/actuals/obsolete/wildcard | ‘structure’ will dump the contents of the JI schema staging tables and error table. ‘actuals’ will dump the contents of the error views that describe the actual staging tables and associated error table. ‘obsolete’ will dump the names of the obsolete retained feature members. ‘wildcard’ will dump the contents of all tables and views that match the wildcard table name. Only non-empty tables are dumped to a file. |
| wildcard table name | Used with ‘wildcard’, defines what tables and views are to be dumped based on Oracle ‘like’ match string syntax. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [output path] [structure/actuals/obsolete/wildcard] [wildcard table name] |
| -9001 | Dump failed |
| 9000 | No records found in any {0} tables. |
| 9001 | Number of {0} table dump files written: {1} |

## Get Input Source Name

**Mid-Tier**: Primary only

**Purpose**: To get the input source name of a Fact Table Family (FTF).

**Usage**: GetInputSourceName [user] [data source] [FTF]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Schema user name |
| data source | TNS entry for the database instance |
| FTF | The name of the Fact Table Family |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [FTF] |
| -9001 | Cannot determine input source name |

## Foundation Restart Status

**Mid-Tier**: Primary only

**Purpose**: To determine if a Foundation command can be started or if it requires a restart/recovery.

**Usage**: FoundationRestartStatus [user] [data source] [command name] [FTF] [type]

[FTF] required only when [command name] is pkbfdatupd or pkbfdatrm.

[type] required only when [command name] is pkbfdatrm.

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| command name | Foundation command name – supported commands are pkbfstrupd, pkbfdatrm, pkbfdatupd and pkbfstrstime |
| FTF | Fact Table Family name – only used with pkbfdatupd and pkbfdatrm |
| type | Either facttables or inputsources – only used with pkbfdatrm |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [command name] [FTF] [type]  [FTF] required only when [command name] is pkbfdatupd or pkbfdatrm.  [type] required only when [command name] is pkbfdatrm. |
| -9001 | Cannot determine if restart is pending |
| 9000 | Restart not pending for command |
| 9001 | Restart pending for command |
| 9002 | Command is currently running |
| 9003 | Restart pending for different type |
| 9004 | Restart pending for blocking command |
| 9005 | Restart pending for different fact table |
| 9006 | Blocking command is currently running |

## Latest Fact Data Time

**Mid-Tier**: Primary

**Purpose**: To either return the Time Member ID of the most recent data in a Fact Table Family (Granite) Table or to determine whether or not fact data had been loaded within the last number of hours specified.

**Usage**: LatestFactDataTime [user] [data source] [ftf table name] [time column] [hours]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Schema user name |
| data source | TNS entry for EKB instance |
| ftf table name | The name of the Fact Table Family (Granite) Table from which the latest time will be queried from |
| time column | The name of the Column in the FTF Table that contains the time Member value to be queried |
| hours | An optional argument that when used results in a return value of either ‘True’ or ‘False’ indicating whether or not the FTF Table had been updated with fact data within the last number of hours specified. |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [ftf table name] [time column] [hours] |
| -9001 | Cannot determine latest fact data time for table:column [*ftf table name:time column*] |

## Process Adhoc Batch (obsolete)

**Mid-Tier**: Primary

**Purpose**: To process the scheduled Jobs associated with a Job Group. Jobs in a Job Group are executed in parallel within their own threads using the [StartProcess](#_Start_Process) command.

This command will create the following tables if they do not already exist:

ADHOC\_BATCH\_QUEUE

ADHOC\_QUEUE\_JOBS

ADHOC\_BATCH\_JOB

The specification of available Job Groups in the ADHOC\_BATCH\_QUEUE table and available Job Names in the ADHOC\_BATCH\_JOB table must be performed using a database administration application such as Toad.

**Usage**: ProcessAdhocBatch [user] [data source] [job group]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Schema user name |
| data source | TNS entry for EKB instance |
| job group | The Job Group who’s associated Jobs will be run |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [job group] |
| -9001 | Cannot complete request for Job Group {0} |
| 9000 | All jobs in Job Group {0} succeeded |
| 9001 | One or more jobs within Job Group {0} failed |

## Quartz Restart Status

**Mid-Tier**: Primary and Secondary

**Purpose**: To determine if a Quartz command can be started or if it requies a restart/recovery.

**Usage**: QuartzRestartStatus [user] [data source] [command name] ['data' | 'structure']

['data' | 'structure'] required only when [command name] is cimsyncpkbf.

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Quartz schema user name |
| data source | TNS entry for EKB instance |
| command name | Quartz command name – supported commands are cimsyncpkbf, cimdatrm and cimdatupd |
| ‘data’ | ‘structure’ | The type of quartz sync – only used with cimsyncpkbf |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [command name] ['data' | 'structure']  ['data' | 'structure'] required only when [command name] is cimsyncpkbf. |
| -9001 | Cannot determine if restart is pending |
| 9000 | Restart not pending for command |
| 9001 | Restart pending for command |
| 9002 | Command is currently running |
| 9003 | Restart pending for different type |
| 9004 | Restart pending for blocking command |
| 9006 | Blocking command is currently running |

## Quartz Sync Status

**Mid-Tier**: Primary and Secondary

**Purpose**: To determine if a Quartz Cube needs to be structure and/or data synchronized.

**Usage**: QuartzSyncStatus [user] [data source] [cube name]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| cube name | Quartz Cube name |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [cube name] |
| -9001 | Cannot determine if a sync is pending |
| 9000 | Quartz Cube structure and data is in sync |
| 9001 | Both structure and data syncs are pending |
| 9002 | Structure sync is pending |
| 9003 | Data sync is pending |

## Row Count

**Mid-Tier**: Primary only

**Purpose**: To get the row count of a database table.

**Usage**: RowCount [user] [data source] [table name] [allowed error codes]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Schema user name |
| data source | TNS entry for the database instance |
| table name | The name of the database table to get the row count from |
| allowed error codes | Comma separated list of error codes used to filter rows based on the value in the “ERROR\_CODE” column. |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [table name] |
| -9001 | Cannot determine row count |
| >= 0 | The tables row count |

## Run SQL

**Mid-Tier**: Primary only

**Purpose**: Runs an SQL statement or PL/SQL against a known datasource without having to provide clear text passwords. If there is a result set, a header is returned followed by data rows delimited with commas.

**Usage**: RunSQL [user] [data source] [sql path]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Foundation schema user name |
| data source | TNS entry for EKB instance |
| sql path | A file containing the SQL or PL/SQL to run. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [sql path] |
| -9001 | Command failed for file [*sql path*] |
| 9000 | Command succeded. Only output is if Oracle returned output. |

## Schedule Adhoc Batch (obsolete)

**Mid-Tier**: Primary

**Purpose**: To add a Job to a Job Group, essentially queueing it for future execution by the [ProcessAdhocBatch](#_Process_Adhoc_Batch) command.

This command will create the following tables if they do not already exist:

ADHOC\_BATCH\_QUEUE

ADHOC\_QUEUE\_JOBS

ADHOC\_BATCH\_JOB

The specification of available Job Groups in the ADHOC\_BATCH\_QUEUE table and available Job Names in the ADHOC\_BATCH\_JOB table must be performed using a database administration application such as Toad.

**Usage**: ScheduleAdhocBatch [user] [data source] [action] [job group] [job name] [once only]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Schema user name |
| data source | TNS entry for EKB instance |
| action | ‘Add’ to add a Job to the Job Group or ‘Remove’ to remove a Job from a Job Group |
| job group | The Job will be added to or removed from this Job Group |
| job name | The Job that will be added to or removed from the Job Group |
| once only | ‘T’ indicates the Job is to be deleted from the Job Group once the Job Group has been processed successfully. ‘F’ indicates the Job should remain in the Job Group after processing. ‘T’ is the default value. |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] [action] [job group] [job name] [once only] |
| -9001 | Cannot complete request for Job Name {0} in Job Group {1} |
| 9000 | Request has been completed for Job Name {0} in Job Group {1} |

## Send REST Request (Mulesoft)

**Mid-Tier**: Primary only

**Purpose**: Sends a web GET, POST, PUT or DELETE request with parameters to an end point and displays the response.

**Usage:** [GET/POST/PUT/DELETE] [end point] parms=[parameters] json=[json path] timeout=[timeout]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| method | One of the following values: GET, POST, PUT and DELETE. Required |
| end point | URL of the API. Required |
| parms | Parameter string to append to the API URL. |
| json | File Path to JSON data to be sent as payload to the API. |
| timeout | Timeout in seconds for the request to complete. 100 is the default value. |

**Return Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [GET/POST/PUT/DELETE] [end point] parms=[parameters] json=[json path] timeout=[timeout] |
| -9001 | Command succeeded for url [end point] |
| 9000 | Command failed for url [end point] |

## Start Process

**Mid-Tier**: Primary and Secondary

**Purpose**: To start one or more processes in the background and wait for the processes to complete.

**Usage**: StartProcess [num concurrent processes] [command line] ... [ruleset xml file]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| num concurrent processes | The number of commands that will be started at a given time. If not specified, all commands will be started at once. Commands are started in the order they are specified on the command line. |
| command line | 1 or more command lines to be started in a background process. If more than one command line, a space character must be used as a delimiter. |
| ruleset xml file | XML file that describes what codes returned by a started process consistute success, failure and partial success. This file is required to ever receive error codes 9002 or 9004. By default, a return code of zero is success and non-zero is failure. If a ruleset does not define success or failure, defaults will apply. |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [num concurrent processes] [command line] ... [ruleset xml file] |
| -9001 | A command failed to start |
| 9000 | All commands completed successfully |
| 9001 | All commands failed |
| 9002 | All commands partially succeeded |
| 9003 | Some commands failed |
| 9004 | Some commands partially succeeded |

**Sample Ruleset XML File:**

In the following example, return codes 0 and 1 constitutes success. Return code 3 defines failure and return code 2 mean partial success. Note that return codes other than 0, 1, 2 and 3 will be defaulted to failure.

<?xml version="1.0"?>

<StartProcess>

<Success>

<ReturnCode>0</ReturnCode>

<ReturnCode>1</ReturnCode>

</Success>

<Failure>

<ReturnCode>3</ReturnCode>

</Failure>

<PartialSuccess>

<ReturnCode>2</ReturnCode>

</PartialSuccess>

</StartProcess>

**Sample Output:**

The Start Process command captures the standard out of each background process. Each output line from a background process is prefixed with its process id in square brackets and written to standard out. When all processes have completed, a summary report documents the process id, command, arguments, start time, end time and return code of each process.

The output of Start Process looks like…

[5792] Restart not pending for command

[5792] EKB Quartz to Foundation Synchronization : Version 7.9.0.1238 for Oracle 11g on WINDOWS - Copyright (c) 1997-2012, JDA Software, Inc.

[5792] Wed Feb 20 18:39:04 2013 (2013-02-20 23:39:04 UT) CIMSYNCPKBF 7248 Info: v7.9.0.1238

[5792] Start processing: cimsyncpkbf / PLAN\_ACTUAL type=structure execute=y setcurrentfoundationtime=y

[3644] Restart not pending for command

[9988] Restart not pending for command

[3644] EKB Quartz to Foundation Synchronization : Version 7.9.0.1238 for Oracle 11g on WINDOWS - Copyright (c) 1997-2012, JDA Software, Inc.

[3644] Wed Feb 20 18:39:05 2013 (2013-02-20 23:39:05 UT) CIMSYNCPKBF 3060 Info: v7.9.0.1238

[3644] Start processing: cimsyncpkbf / PLAN\_ACTUAL\_ALLOC type=structure execute=y setcurrentfoundationtime=y

[9988] EKB Quartz to Foundation Synchronization : Version 7.9.0.1238 for Oracle 11g on WINDOWS - Copyright (c) 1997-2012, JDA Software, Inc.

[9988] Wed Feb 20 18:39:05 2013 (2013-02-20 23:39:05 UT) CIMSYNCPKBF 676 Info: v7.9.0.1238

[9988] Start processing: cimsyncpkbf / PLAN\_CENTRAL type=structure execute=y setcurrentfoundationtime=y

[5792] Trace output will be appended to \\Wdcae-dpln11p\Quartz\plan\_actual\log\CIMSyncPKBF\_STRUCTURE.trc

[5792] Wed Feb 20 18:39:06 2013 (2013-02-20 23:39:06 UT) CIMSYNCPKBF 7248 Info:

[5792] Starting job STRUCTURE to synchronize PLAN\_ACTUAL with current Foundation structure

[3644] Trace output will be appended to \\Wdcae-dpln11p\Quartz\plan\_actual\_alloc\log\CIMSyncPKBF\_STRUCTURE.trc

[9988] Trace output will be appended to \\Wdcae-dpln11p\Quartz\plan\_central\log\CIMSyncPKBF\_STRUCTURE.trc

[3644] Wed Feb 20 18:39:07 2013 (2013-02-20 23:39:07 UT) CIMSYNCPKBF 3060 Info:

[3644] Starting job STRUCTURE to synchronize PLAN\_ACTUAL\_ALLOC with current Foundation structure

[9988] Wed Feb 20 18:39:07 2013 (2013-02-20 23:39:07 UT) CIMSYNCPKBF 676 Info:

[9988] Starting job STRUCTURE to synchronize PLAN\_CENTRAL with current Foundation structure

[5792] Wed Feb 20 18:39:12 2013 (2013-02-20 23:39:12 UT) CIMSYNCPKBF 7248 Info:

[5792] Job STRUCTURE completed

[5792] Wed Feb 20 18:39:12 2013 (2013-02-20 23:39:12 UT) CIMSYNCPKBF 7248 Info:

[5792] Status: Process completed successfully

[3644] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 3060 Info:

[3644] Job STRUCTURE completed

[3644] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 3060 Info:

[3644] Status: Process completed successfully

[5792] EKB Quartz to Foundation Synchronization : Version 7.9.0.1238 for Oracle 11g on WINDOWS - Copyright (c) 1997-2012, JDA Software, Inc.

[5792] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 8680 Info: v7.9.0.1238

[5792] Start processing: cimsyncpkbf / PLAN\_ACTUAL type=data execute=y

[9988] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 676 Info:

[9988] Job STRUCTURE completed

[9988] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 676 Info:

[9988] Status: Process completed successfully

[3644] EKB Quartz to Foundation Synchronization : Version 7.9.0.1238 for Oracle 11g on WINDOWS - Copyright (c) 1997-2012, JDA Software, Inc.

[3644] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 6676 Info: v7.9.0.1238

[3644] Start processing: cimsyncpkbf / PLAN\_ACTUAL\_ALLOC type=data execute=y

[9988] EKB Quartz to Foundation Synchronization : Version 7.9.0.1238 for Oracle 11g on WINDOWS - Copyright (c) 1997-2012, JDA Software, Inc.

[9988] Wed Feb 20 18:39:13 2013 (2013-02-20 23:39:13 UT) CIMSYNCPKBF 6528 Info: v7.9.0.1238

[9988] Start processing: cimsyncpkbf / PLAN\_CENTRAL type=data execute=y

[5792] Trace output will be appended to \\Wdcae-dpln11p\Quartz\plan\_actual\log\CIMSyncPKBF\_DATA.trc

[3644] Trace output will be appended to \\Wdcae-dpln11p\Quartz\plan\_actual\_alloc\log\CIMSyncPKBF\_DATA.trc

[9988] Trace output will be appended to \\Wdcae-dpln11p\Quartz\plan\_central\log\CIMSyncPKBF\_DATA.trc

[9988] Wed Feb 20 18:39:16 2013 (2013-02-20 23:39:16 UT) CIMSYNCPKBF 6528 Info:

[9988] Starting job DATA to synchronize PLAN\_CENTRAL with current Foundation data

[5792] Wed Feb 20 18:39:16 2013 (2013-02-20 23:39:16 UT) CIMSYNCPKBF 8680 Info:

[5792] Starting job DATA to synchronize PLAN\_ACTUAL with current Foundation data

[3644] Wed Feb 20 18:39:16 2013 (2013-02-20 23:39:16 UT) CIMSYNCPKBF 6676 Info:

[3644] Starting job DATA to synchronize PLAN\_ACTUAL\_ALLOC with current Foundation data

[9988] Wed Feb 20 18:39:38 2013 (2013-02-20 23:39:38 UT) CIMSYNCPKBF 6528 Info:

[9988] Job DATA completed

[9988] Wed Feb 20 18:39:38 2013 (2013-02-20 23:39:38 UT) CIMSYNCPKBF 6528 Info:

[9988] Status: Process completed successfully

[3644] Wed Feb 20 18:43:25 2013 (2013-02-20 23:43:25 UT) CIMSYNCPKBF 6676 Info:

[3644] Job DATA completed

[3644] Wed Feb 20 18:43:25 2013 (2013-02-20 23:43:25 UT) CIMSYNCPKBF 6676 Info:

[3644] Status: Process completed successfully

[5792] Wed Feb 20 18:49:30 2013 (2013-02-20 23:49:30 UT) CIMSYNCPKBF 8680 Info:

[5792] Job DATA completed

[5792] Wed Feb 20 18:49:30 2013 (2013-02-20 23:49:30 UT) CIMSYNCPKBF 8680 Info:

[5792] Status: Process completed successfully

PID: 9988

Command: CubeSync.bat

Arguments: plan\_central PLAN\_CENTRAL

Started: 2/20/2013 6:39 PM

Ended: 2/20/2013 6:39 PM

Return Code: 0

PID: 3644

Command: CubeSync.bat

Arguments: plan\_actual\_alloc PLAN\_ACTUAL\_ALLOC

Started: 2/20/2013 6:39 PM

Ended: 2/20/2013 6:43 PM

Return Code: 0

PID: 5792

Command: CubeSync.bat

Arguments: plan\_actual PLAN\_ACTUAL

Started: 2/20/2013 6:39 PM

Ended: 2/20/2013 6:49 PM

Return Code: 0

All commands completed successfully

## Truncate JI Errors (Mulesoft & Datastage)

**Mid-Tier**: Primary only

**Purpose**: To delete all rows from the EXT\_EPSD\_ERRORS table in the JI Schema.

**Usage**: TruncateJIErrors [user] [data source]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | JI schema user name |
| data source | TNS entry for the EKB instance |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Error Message** |
| -9002 | Usage: [user] [data source] |
| -9001 | Cannot truncate JI error table |

## Encrypted Password Repository

Most EP Batch Facility commands require database login credentials. While the user names are specified as command line arguments, the passwords associated with these users must be defined in the EPBatchFacility.xml file.

When passwords are specified or changed, they are entered into the EPBatchFacility.xml file as clear text and an encoding type of none. Example:

<?xml version="1.0"?>

<EPBatchFacility>

<Credentials>

<Credential>

<ID>ext</ID>

<Password>ext</Password>

<Encoding>none</Encoding>

</Credential>

</Credentials>

</EPBatchFacility>

When an EP Batch Facility console command is run, all clear text passwords will be encrypted and the encoding type is set to AES (indicating the Advanced Encryption Standard). Example after encryption has been performed:

<?xml version="1.0"?>

<EPBatchFacility>

<Credentials>

<Credential>

<ID>ext</ID>

<Password>LKCFIPcwKdc6MdS6uXJ5/Q==</Password>

<Encoding>AES</Encoding>

</Credential>

</Credentials>

</EPBatchFacility>

# EKBF Server Commands Section 4

These commands are fully documented in the EP Server Administrator Guide. Excerpts from that documented are provided below.

## Foundation Data Remove (FDR)

The Foundation Data Remove (FDR) server process is an EKBF server-based command-line program typically run on a regular schedule. The FDR server process performs the following:

* Enacts the data life rules defined in metadata for the fact table families by removing records that are older than the specified data life.
* Purges rows from input source tables that have already been loaded by Foundation Data Update (FDU) for the input sources that are maintained by Foundation or have been designated for purging by EKB Facts.
* Purges rows from Granite and Slate tables for dimension members that have been deleted.
* Purges rows from Granite and Slate tables where all the measure values are NULL.
* Drops columns for deleted fact table family measures. These columns were marked unused by Foundation Structure Update when the measures were deleted from the fact table family or from an input source or Slate table.

## Foundation Data Update (FDU)

The Foundation Data Update server process is an EKBF server command-line process that takes fact data from an input source (database tables) and moves it into a Foundation fact table family (FTF). Because FTFs can include stored consolidations, the Foundation Data Update process consolidates input data for update to the tables holding the consolidations (Slate tables). This server process loads facts from source applications including the clients' host system and the Enterprise Planning client application.

## FDU Transaction History Utility

The Foundation Data Update server process keeps a history of the transaction sets (range of transaction IDs) that are successfully loaded in Foundation. The history of successfully loaded transaction sets includes the input source name, the minimum and maximum transaction IDs, the transaction record count, and the update date for each set. The Foundation Data Carry Forward Balance server process also keeps a history of the periods it populates and the latest update date for each run. The FDU Transaction History Utility allows you to display the transaction history and carry forward balance history for one or more fact table families or input sources.

The FDU Transaction History Utility is used periodically to prune the historical record of the successfully loaded transaction sets and carry forward balance runs. The FDU Transaction History Utility does not delete input source or fact table data. When deleting from the transaction history, the "high water mark" is never deleted. The only exception is if there are rows in the transaction history that are associated with input sources that are no longer defined. The FDU Transaction History Utility deletes those rows, if requested.

The "high water mark" mentioned previously is used currently by many Foundation processes, as well as other applications for different purposes.

* Foundation Data Update uses the high water mark to determine the next transaction set to load and to ensure that it does not load a transaction set twice.
* Foundation Data Remove uses the high water mark to ensure that it does not purge input source table rows that have not yet been loaded.
* Foundation Data Export server process and stored procedures use the high water mark to read unposted fact data when retrieving Latest facts using Foundation Data Export.

## Foundation Set Time (FST)

Foundation includes the concept of "current time". Current time in Foundation is used in a number of ways.

* It controls data aging.
* It is used to perform posting rule validation.
* It allows users to see the correct values for measures with Closing consolidation type in, for example, the current season when you are only halfway through.

Current time comprises a designated member in the lowest level of the primary Calendar dimension and its ancestors in the primary Calendar dimension and any connected Secondary dimensions. For example, in a four-level Calendar hierarchy with levels DAY, WEEK, PERIOD, and YEAR, current time comprises the current DAY member and its WEEK, PERIOD, and YEAR ancestors. When Calendar members are first added to the Foundation database by Foundation Structure Update, Current time is set to the earliest DAY member loaded and its ancestors. Use the Foundation Set Time server process to change this initial setting and update it on a regular basis.

The Foundation Set Time server process writes to production metadata tables and takes effect immediately.

## Foundation Structure Update (FSU)

Foundation Structure Update (FSU) is an EKBF server command-line process that is run on a regular schedule to take dimension member changes from a staging area in EKB Structure and process them into the dimension tables held in Foundation. During the update, the process populates staging tables for structure update to the Quartz cubes. Foundation Structure Update also implements metadata changes made using Architect for Foundation.

## Quartz Data Remove (QDR)

The Quartz Data Remove server process lets you manage the retention of fact table data in your cube.

The Quartz Data Remove server process removes data that is no longer of business interest according to your data aging rules. We recommend matching your aging rules in Quartz with the aging rules for the corresponding fact table families in Foundation. When aging a fact table that contains Latest or Closing-type measures, Quartz Data Remove automatically retains data in one extra period prior to the first period to be retained according to the aging rule so that measure derivations that compute opening balance values from the next period closing balance value will result in a value for the earliest period to be retained.

## Quartz Sync Foundation (Quartz Structure Sync/Quartz Data Sync)

Quartz cubes are synchronized with the Foundation database using the Quartz to Foundation Synchronization (cimsyncpkbf) server process.

The synchronization procedures typically run on a scheduled basis and are facilitated by the use of staging tables and views. Foundation server processes populate Quartz structure staging tables to drive structure synchronization and create views to drive fact synchronizations. Then, Quartz server processes apply the contents of these tables and views to the Quartz cubes.

The Quartz to Foundation synchronization process also releases metadata changes made through Architect for Quartz and the Quartz Structure Update server process.

Quartz to Foundation Synchronization locks out most other Quartz processes for the duration of the run. Depending on the command line options, during the Release phase of the synchronization it may also lock out applications that read from the Quartz cube.

## EP Mid-Tier Admin Client Console

There are important EP Mid-Tier processes that can be run from the command line to help support the batch processing and operational process flow.

The EP Mid-Tier Console application is installed with the EP Mid-Tier Administrator. It can be found in the JDA\EPMidTierAdmin\bindirectory. It processes command line input and notifies the proper component of an EP Mid-Tier to execute the specified process.

Processes Supported by the Console application:

* Rebuild the Master Measure Map (MMM).
* Rebuild an EKB Context.

# Batch Scripts Section 5

## Actuals Cube Remove Data

**Mid-Tier**: Primary and Secondary

**Purpose**: Multithreads the execution of multiple Quartz Cube data removes.

**Usage**: ActualsCubeRemoveData

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Actuals Cube Sync

**Mid-Tier**: Primary and Secondary

**Purpose**: Multithreads the execution of multiple Quartz Cube structure and data syncs. Quartz Cubes are those that contain Actuals data only.

**Usage**: ActualsCubeSync

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Actuals Data Update

**Mid-Tier**: Primary only

**Purpose**: Executes EKBF Commands to load fact data into Granite for a single Actuals FTF and then optionally purges the contents of the staging table. If there are errors in the processing of data, the staging table will not be purged.

**Usage**: ActualsDataUpdate [FTF] [retain/truncate]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| FTF | Fact Table Family name |
| retain/truncate | Specifying ‘retain’ will disable the truncating of input source tables except in restart scenarios. The default is ‘truncate’ which does not have to be specified. |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |
| 9999 | Errors in staged data |

**Server Command Lines:**

[pkbfdatrm](#_Foundation_Data_Remove) / type=facttables ftf=ACTUAL\_ON\_ORDER analyze=n empty=y

[pkbfdatrm](#_Foundation_Data_Remove) / type=facttables ftf=ACTUAL\_ON\_ORDER analyze=n empty=y restart=s

[pkbfdatupd](#_Foundation_Plan_Data) / *[FTF]* continueiferrors=y bulkloadgranite=%BULKLOAD% dropindexgranite=y

[pkbfdatupd](#_Foundation_Data_Update) / *[FTF]* continueiferrors=y bulkloadgranite=%BULKLOAD% dropindexgranite=y restart=y

*%BULKLOAD% is ‘Y’ for ACTUAL\_ON\_ORDER and ‘N’ for all other FTFs*

[pkbfdatrm](#_Foundation_Data_Remove) / type=inputsources ftf=*[FTF]* analyze=n empty=y

[pkbfdatrm](#_Foundation_Data_Remove) / type=inputsources ftf=*[FTF]* analyze=n empty=y restart=s

**Workflow:**



## Allow Client Connections

**Mid-Tier**: Primary only

**Purpose**: Creates or deletes a trigger file used by the EP Client Shutdown functionality to determine if EP Client connections should be allowed or disallowed.

**Usage**: AllowClientConnections [enable/disable]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| enable/disable | Inidicates whether client connections are enabled or disabled |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

## Archive Logs

**Mid-Tier**: Primary and Secondary

**Purpose**: Creates a subdirectory inside the Archive of the batch working directory (See [EP Batch Client and EP Batch Service](#_EP_Batch_Client)) for the current date and moves working files to the directory. Also, purges aged log archives.

See [GlobalVariables.bat](#_Global_Variables_1) for configuring source directories and file extensions that ArchiveLogs uses when performing file moves.

**Usage**: ArchiveLogs

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

## Clear Import Tables (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Clear the Import Tables in the EKB Schema.

**Usage**: ClearImportTables

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) POST %CLEARIMPORTTABLESURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Create AG Tables (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Create the AG Tables in the EXT Schema.

**Usage**: CreateAGTables

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) POST %CREATEAGTABLESURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Cube Purge Aged Data

**Mid-Tier**: Primary and Secondary

**Purpose**: Executes EKBF Command to purge aged data on a single Quartz Cube.

**Usage**: CubePurgeAgedData [user] [quartz db]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Quartz schema user name |
| quartz db | Quartz schema name |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[cimdatrm](#_Quartz_Data_Remove) / *[quartz db]* empty=n

[cimdatrm](#_Quartz_Sync_Foundation) / *[quartz db]* empty=n restart=y

**Workflow:**



## Cube Remove Data

**Mid-Tier**: Primary and Secondary

**Purpose**: Executes EKBF Commands to remove data from a Quartz Cube

**Usage**: CubeRemoveData [user] [quartz db]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Quartz schema user name |
| quartz db | Quartz schema name |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[cimdatrm](#_Quartz_Data_Remove) / *[quartz db]* empty=y all=y confirmempty=n

[cimdatrm](#_Quartz_Sync_Foundation) / *[quartz db]* empty=y all=y confirmempty=n restart=y

**Workflow:**



## Cube Sync

**Mid-Tier**: Primary and Secondary

**Purpose**: Executes EKBF Commands to perform a structure and data sync on a single Quartz Cube.

**Usage**: CubeSync [user] [quartz db]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| user | Quartz schema user name |
| quartz db | Quartz schema name |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[cimsyncpkbf](#_Quartz_Sync_Foundation) / *[quartz db]* type=structure

[cimsyncpkbf](#_Quartz_Sync_Foundation) / *[quartz db]* type=structure restart=y

[cimsyncpkbf](#_Quartz_Sync_Foundation) / *[quartz db]* type=data

[cimsyncpkbf](#_Quartz_Sync_Foundation) / *[quartz db]* type=data restart=y

**Workflow:**



## EKB Cleanup

**Mid-Tier**: Primary only

**Purpose**: Performs general cleanup activities.

**Usage**: EKBCleanup

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[pkbfdutrans](#_FDU_Transaction_History) / delete confirmdelete=n

## FDE Data Update

**Mid-Tier**: Primary only

**Purpose**: Multithreads the execution of multiple fact data loads into Granite for data previously extracted and staged using the [CopyFactData](#_Copy_Fact_Data) console application.

**Usage**: FDEDataUpdate

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## FDE Stage Fact Data

**Mid-Tier**: Primary only

**Purpose**: Multithreads the execution of multiple FDE extracts and input source table loads.

**Usage**: FDEStageFactData

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Foundation Actuals Data Update

**Mid-Tier**: Primary only

**Purpose**: Multithreads the execution of multiple fact data loads into Granite for Actuals FTFs.

**Usage**: FoundationActualsDataUpdate

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Foundation Plan Data Update

**Mid-Tier**: Primary only

**Purpose**: Multithreads the execution of multiple fact data loads into Granite for User Plan FTFs.

**Usage**: FoundationPlanDataUpdate

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Foundation Purge Aged Data

**Mid-Tier**: Primary only

**Purpose**: Purges aged data from Foundation.

**Usage**: FoundationPurgeAgedData

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[pkbfdatrm](#_Foundation_Data_Remove) / type=FACTTABLES empty=n

[pkbfdatrm](#_Foundation_Data_Remove) / type=FACTTABLES empty=n restart=s

**Workflow:**



## Foundation Set Time

**Mid-Tier**: Primary only

**Purpose**: Sets the Foundation Current Time. If an increment, Calendar Member ID or “latest” are not specified as arguments, the increment is taken from the [DEFAULTINCREMENT](#_Global_Variables) global variable.

**Usage**: FoundationSetTime [increment | calendar member id | “latest”]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| increment | calendar member id | “latest” | EP supported increment such as ‘+week’, a Calendar Member ID from EKBS or the calendar member id returned from a call to [LatestFactDataTime](#_Latest_Fact_Data) |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[pkbfstrstime](#_Foundation_Set_Time) / *%DEFAULTINCREMENT%*

[pkbfstrstime](#_Foundation_Set_Time) / *%DEFAULTINCREMENT%* restart=y

[pkbfstrstime](#_Foundation_Set_Time) / *[increment | calendar member id]*

[pkbfstrstime](#_Foundation_Set_Time) / *[increment | calendar member id]* restart=y

**Workflow:**



## Foundation Structure Update

**Mid-Tier**: Primary only

**Purpose**: Synchronizes EKB Foundation with EKB Structure.

**Usage**: FoundationStructureUpdate

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Server Command Lines:**

[pkbfstrupd](#_Global_Variables) /

[pkbfstrupd](#_Global_Variables) / restart=y

**Workflow:**



## Gather Schema Stats

**Mid-Tier**: Primary

**Purpose**: Performs an Oracle Gather Schema Stats based on an SQL file.

**Usage**: GatherSchemaStats

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



#### Sample GatherSchemaStats.sql

BEGIN

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'ekbf');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'ext');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_actual');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_bulp');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_bupp');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_molp');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_mopp');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_stpp');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_tdlp');

DBMS\_STATS.GATHER\_SCHEMA\_STATS(ownname => 'plan\_tdpp');

END;

## Get Batch Result (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Get Result for the Specified Batch.

**Usage**: GetBatchResult [batch id]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| Batch id | The ID of the Structure Import Batch instance |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) GET %GETBATCHRESULTURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Get EKB Schema Config (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Get EKB Schema Configuration.

**Usage**: GetEKBSchemaConfig

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) GET %GETEKBSCHEMACONFIGURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Get EXT Schema Config (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Get EXT Schema Configuration.

**Usage**: GetEXTSchemaConfig

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) GET %GETEXTSCHEMACONFIGURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Global Variables

**Mid-Tier**: Primary and Secondary

**Purpose**: Single location to set variable values used by various other batch scripts.

**Usage**: GlobalVariables

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Global Variables**:

|  |  |
| --- | --- |
| **GLOBAL SCRIPT VARIABLES** | |
| **Variable** | **Sample Value** |
| DATASOURCE | epdev |
| FOUNDATIONUSER | ekbf |

|  |  |
| --- | --- |
| **JI SCRIPT VARIABLES** | |
| **Variable** | **Sample Value** |
| MULE | Y |
| JIUSER | ext |

|  |  |
| --- | --- |
| **JI ON MULE ONLY SCRIPT VARIABLES** | |
| **Variable** | **Sample Value** |
| MULEENV | dev |
| MULELABEL | defaultLabel |
| RESTTIMEOUT | 100 |
| CLEARIMPORTTABLESURL | http://localhost:9091/clear/ekbsimporttables |
| CREATEAGTABLESURL | http://localhost:9091/init/agtables |
| GETBATCHRESULTURL | http://localhost:9091/result/ |
| GETEKBSCHEMACONFIGURL | http://localhost:9091/config/ep |
| GETEXTSCHEMACONFIGURL | http://localhost:9091/config/ji |
| INITEXTSCHEMAURL | http://localhost:9091/init/extschema |
| PENDINGBATCHPROCESSESURL | http://localhost:9091/import/pendingbatches |
| PROCESSBERELDATAURL | http://localhost:9091/import/crossdimrel |
| PROCESSMEMBERDATAURL | http://localhost:9091/import/memberdata |
| PROCESSMETADATAURL | http://localhost:9091/import/metadata |
| SETJIPROPERTIESURL | http://localhost:9811/cps/api/v1/application/ji/properties |
| EKBSCHEMAPROPS | "?categories=%FOUNDATIONUSER%&environment=%MULEENV%&label=%MULELABEL%" |
| EMAILPROPS | "?categories=email&environment=%MULEENV%&label=%MULELABEL%" |
| EXTSCHEMAPROPS | "?categories=%JIUSER%&environment=%MULEENV%&label=%MULELABEL%" |
| FRAMEWORKPROPS | "?categories=eFramework&environment=%MULEENV%&label=%MULELABEL%" |
| HTTPPROPS | "?categories=http&environment=%MULEENV%&label=%MULELABEL%" |
| JIPROPS | "?environment=%MULEENV%&label=%MULELABEL%" |

|  |  |
| --- | --- |
| **JI ON DATASTAGE ONLY SCRIPT VARIABLES** | |
| **Variable** | **Sample Value** |
| DSDIR | D:\IBM\InformationServer\Clients\Classic\ |
| DSPASSWORD | isadmin |
| DSPROJECT | SPEXTEPSD |
| DSSERVER | WDCAE-DPLN11P |
| DSUSER | isadmin |

|  |  |
| --- | --- |
| **MID-TIER STOP START VARIABLES** | |
| **Variable** | **Sample Value** |
| PRISEC | primary |
| SERVERCLONES | JDABIClonedServer\_PrimaryClone1,JDABIClonedServer\_PrimaryClone2 |

|  |  |
| --- | --- |
| **CONTEXT REBUILD VARIABLES** | |
| **Variable** | **Sample Value** |
| CONTEXTNAMES | EPPlan |

|  |  |
| --- | --- |
| **FOUNDATON TIME VARIABLES** | |
| **Variable** | **Sample Value** |
| DEFAULTINCREMENT | +WEEK |
| DEFAULTFTFTABLE | PF\_FT\_ACTUAL\_RECEIPTS\_000 |
| DEFAULTFTFTIMECOLUMN | CALENDAR\_KEY |

|  |  |
| --- | --- |
| **ACTUAL DATA POSTING VARIABLES** | |
| **Variable** | **Sample Value** |
| ALLOWEDERRORCODES | 23 |
| FORCEEMPTY | ACTUAL\_ON\_ORDER |
| FULLSTATEMENT | ACTUAL\_ON\_ORDER |

|  |  |
| --- | --- |
| **EP SHUTDOWN VARIABLES** | |
| **Variable** | **Sample Value** |
| TRIGGERPATH | \\[admin sever]\Toolkit\Corporate\Plan\trigger.ini |

|  |  |
| --- | --- |
| **LOG ARCHIVING VARIABLES** | |
| **Variable** | **Sample Value** |
| ARCHIVEFILEEXTENSIONS | log,trc,errlist,errors,infolist,bin,shape,facttables,sr,tmp,merged,sts,rsp,tin,al,pl,sorted |
| EXPIREDAYS | 28 |
| QUARTZHOMEPATHS | D:\Quartz,C:\ProgramData\JDA\EKBF |

|  |  |
| --- | --- |
| **CLOUD SFTP VARIABLES** | |
| **Variable** | **Sample Value** |
| CLOUD | N |
| SFTPPATH | D:\SFTP |
| FACTHISTORYPATH | %SFTPPATH%\Fact\_History |
| INBOXPATH | %SFTPPATH%\Inbox |
| INBOXARCHIVEPATH | %INBOXPATH%\Archive |
| OUTBOXPATH | %SFTPPATH%\Outbox |
| RECLASSPATH | %INBOXPATH%\Reclass |
| SFTPSUPPORTPATH | D:\EP\_Batch\SFTP\_Support |
| SQLLDRCONTROLPATH | %SFTPSUPPORTPATH%\Ctl\_Files |
| SQLLDRERRORSPATH | %SFTPPATH%\SQLLDR\_Errors |
| SQLLDRLOGSPATH | %SFTPPATH%\SQLLDR\_Logs |
| STRUCTURELOADERRORSPATH | %SFTPPATH%\Structure\_Load\_Errors |
| TYLOADERRORSPATH | %SFTPPATH%\TY\_Load\_Errors |

## Init EXT Schema (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Initialize the EXT Schema.

**Usage**: InitEXTSchema

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) POST %INITEXTSCHEMAURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Initialize SFTP (Cloud)

**Mid-Tier**: Primary only

**Purpose**: Initializes the SFTP environment for an EP Cloud implementation.

**Usage**: InitializeSFTP

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## JI Member Data Load (Mulesoft & DataStage)

**Mid-Tier**: Primary only

**Purpose**: Executes the JI Server Data job to load EKBS Member updates.

If %MULE% global variable is ‘Y’ then JI for Mule APIs will be called, otherwise JI for DataStage jobs will be called.

**Usage**: JIMemberDataLoad

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |
| 9999 | Errors in staged data |

**Workflow:**



## JI Meta Data Load (Mulesoft & Datastage)

**Mid-Tier**: Primary only

**Purpose**: Executes the JI Meta Data job to load EKBS Meta Data changes.

If %MULE% global variable is ‘Y’ then JI for Mule APIs will be called, otherwise JI for DataStage jobs will be called.

**Usage**: JIMetaDataLoad

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |
| 9999 | Errors in staged data |

**Workflow:**



## Pending Batch Processes (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Get Pending Batch Processes Info.

**Usage**: PendingBatchProcesses

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) POST %PENDINGBATCHPROCESSESURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Plan Cube Sync

**Mid-Tier**: Primary and Secondary

**Purpose**: Multithreads the execution of multiple Quartz Cube structure and data syncs. Quartz Cubes are those that contain User Plan data only.

**Usage**: PlanCubeSync

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Plan Data Update

**Mid-Tier**: Primary only

**Purpose**: Executes EKBF Commands to load fact data into Granite for a single User Plan FTF and then optionally purges the contents of the staging table. If there are errors in the processing of data, the staging table will not be purged.

**Usage**: PlanDataUpdate [FTF] [retain/truncate]

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| FTF | Fact Table Family name |
| retain/truncate | Specifying ‘retain’ will disable the truncating of input source tables except in restart scenarios. The default is ‘truncate’ which does not have to be specified. |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |
| 9999 | Errors in staged data |

**Server Command Lines:**

[pkbfdatupd](#_Foundation_Plan_Data) / *[FTF]* dropindexgranite=y

[pkbfdatupd](#_Foundation_Data_Update) / *[FTF]* dropindexgranite=y restart=y

[pkbfdatrm](#_Foundation_Data_Remove) / type=inputsources ftf=*[FTF]* analyze=n empty=y

[pkbfdatrm](#_Foundation_Data_Remove) / type=inputsources ftf=*[FTF]* analyze=n empty=y restart=s

**Workflow:**



## Process Member Data (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Requests to Process the AG Member Data Staging Tables and Staged Business Entity Relationships.

**Usage**: ProcessMemberData

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Lines:**

[SendRESTRequest](#_Send_REST_Request) POST %PROCESSMEMBERDATAURL% timeout=%RESTTIMEOUT%

[SendRESTRequest](#_Send_REST_Request) POST %PROCESSBERELDATAURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Process Meta Data (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Request to Process the EXT Meta Data Staging Tables.

**Usage**: ProcessMetaData

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Line:**

[SendRESTRequest](#_Send_REST_Request) POST %PROCESSMETADATAURL% timeout=%RESTTIMEOUT%

**Workflow:**



## Quartz Cube Sync

**Mid-Tier**: Primary and Secondary

**Purpose**: Multithreads the execution of multiple Quartz Cube structure and data syncs. This script includes both Actuals and User Plan Quartz Cubes.

**Usage**: QuartzCubeSync

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Quartz Purge Aged Data

**Mid-Tier**: Primary and Secondary

**Purpose**: Multithreads the execution of multiple Quartz Cube aged data purge processes.

**Usage**: QuartzPurgeAgedData

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Rebuild Contexts

**Mid-Tier**: Primary only

**Purpose**: Executes EP Mid-Tier command to rebuild all Contexts.

**Usage**: RebuildContexts

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Set JI Mule Properties (Mulesoft)

**Mid-Tier**: Primary

**Purpose**: Executes the Mule Requests to Set JI on Mule Property Values.

**Usage**: SetJIMuleProperties

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Send REST Request Command Lines:**

[SendRESTRequest](#_Send_REST_Request) POST %SETJIPROPERTIESURL% parms=%EKBSCHEMAPROPS% json=SetEKBSchemaProperties.json timeout=%RESTTIMEOUT%

[SendRESTRequest](#_Send_REST_Request) POST %SETJIPROPERTIESURL% parms=%EMAILPROPS% json=SetEmailProperties.json

[SendRESTRequest](#_Send_REST_Request) POST %SETJIPROPERTIESURL% parms=%EXTSCHEMAPROPS% json=SetEXTSchemaProperties.json timeout=%RESTTIMEOUT%

[SendRESTRequest](#_Send_REST_Request) POST %SETJIPROPERTIESURL% parms=%FRAMEWORKPROPS% json=SetFrameworkProperties.json timeout=%RESTTIMEOUT%

[SendRESTRequest](#_Send_REST_Request) POST %SETJIPROPERTIESURL% parms=%HTTPPROPS% json=SetHTTPProperties.json timeout=%RESTTIMEOUT%

[SendRESTRequest](#_Send_REST_Request) POST %SETJIPROPERTIESURL% parms=%JIPROPS% json=SetJIProperties.json timeout=%RESTTIMEOUT%

**Workflow:**



#### Sample SetEKBSchemaProperties.json

{

"properties":{

"ep.ekbcsServer":"localhost",

"ep.soapServer":"localhost",

"ep.defaultCulture":"en-US",

"ep.db.host":"localhost",

"ep.db.port":"1521",

"ep.db.sid":"[database SID]",

"ep.db.driverClassName":"oracle.jdbc.OracleDriver",

"ep.db.driver":"jdbc:oracle:thin",

"ep.db.user":"ekb",

"ep.db.password-secure":"ekb"

}

}

#### Sample SetEmailProperties.json

{

"properties":{

"email.enable":"false",

"email.emailsmtp.host":"smtp.jda.com",

"email.emailsmtp.port":"587",

"email.emailsmtp.user":"",

"email.emailsmtp.password-secure":"",

"email.emailaddress.fromaddress":"",

"email.emailaddress.toaddress":""

}

}

#### Sample SetEXTSchemaProperties.json

{

"properties":{

"AGTableScriptFiles.location":"C:\JIGeneratedsqls",

"metadata.databasefileslocation":"C:\JI\IVK\_Metadata\_databaseFiles",

"metadata.ext\_epsd\_import\_attr\_Def.source":"database",

"metadata.ext\_epsd\_import\_attr\_Def.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_attr\_Def.load":"true",

"metadata.ext\_epsd\_import\_attr\_def\_pot.source":"database",

"metadata.ext\_epsd\_import\_attr\_def\_pot.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_attr\_def\_pot.load":"true",

"metadata.ext\_epsd\_import\_hier\_level\_def.source":"database",

"metadata.ext\_epsd\_import\_hier\_level\_def.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_hier\_level\_def.load":"true",

"metadata.ext\_epsd\_import\_hierarchy\_def.source":"database",

"metadata.ext\_epsd\_import\_hierarchy\_def.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_hierarchy\_def.load":"true",

"metadata.ext\_epsd\_import\_level\_def.source":"database",

"metadata.ext\_epsd\_import\_level\_def.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_level\_def.load":"true",

"metadata.ext\_epsd\_import\_mbr\_attr\_def.source":"database",

"metadata.ext\_epsd\_import\_mbr\_attr\_def.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_mbr\_attr\_def.load":"true",

"metadata.ext\_epsd\_import\_member\_def.source":"database",

"metadata.ext\_epsd\_import\_member\_def.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_member\_def.load":"true",

"metadata.ext\_epsd\_import\_well\_known\_lvl.source":"database",

"metadata.ext\_epsd\_import\_well\_known\_lvl.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_well\_known\_lvl.load":"true",

"metadata.ext\_epsd\_import\_rel.source":"database",

"metadata.ext\_epsd\_import\_rel.dbloadtype":"ClearandLoad",

"metadata.ext\_epsd\_import\_rel.load":"true",

"memberrelation.location":"C\JI\IVK\_Metadata\_databaseFiles\EXT\_EPSD\_IMPORT\_MEMBER\_REL.database",

"memberrelation.source":"database",

"memberrelation.dbloadtype":"ClearandLoad"

}

}

#### Sample SetFrameworkProperties.json

{

"properties":{

"eframework.applicationId":"JI for EP",

"eframework.errorReport.enabled":"false",

"eframework.errorReport.htmlTemplatePath":"errorReportTemplate.html",

"eframework.errorReport.errorReportPath":"",

"eframework.errorReport.title":"JI 1.0 Error report",

"eframework.errorReport.payload.transaction.size":"2048"

}

}

#### Sample SetHTTPProperties.json

{

"properties":{

"server.listener.config":"HTTP\_Listener\_config",

"server.http.host":"0.0.0.0",

"server.http.port":"9091",

"server.https.host":"0.0.0.0",

"server.https.port":"9092",

"server.https.tls.keyStore.path":"keystore.jks",

"server.https.tls.keyStore.password-secure":"changeme",

"server.https.tls.keyStore.key.password-secure":"changeme"

}

}

#### Sample SetJIProperties.json

{

"properties":{

"ji.db.host":"localhost",

"ji.db.port":"1521",

"ji.db.sid":"[database SID]",

"ji.db.driverClassName":"oracle.jdbc.OracleDriver",

"ji.db.driver":"jdbc:oracle:thin",

"ji.db.user":"ext",

"ji.db.password-secure":"ext"

}

}

## Stage Facts (Cloud)

**Mid-Tier**: Primary

**Purpose**: Stages Fact data from the SFTP Inbox into Foundation Staging tables in the EKBF Schema.

**Usage**: StageFacts

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| -3 | Error processing one or more files |
| -2 | Control file did not arrive in time |
| -1 | Invalid arguments passed |
| 1 | Records were rejected but fewer than the threshold for failure |

## Stage Structure (Cloud)

**Mid-Tier**: Primary

**Purpose**: Stages Structure data from the SFTP Inbox into Structure Staging tables in the EXT Schema.

**Usage**: StageStructure

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| -3 | Error processing one or more files |
| -2 | Control file did not arrive in time |
| -1 | Invalid arguments passed |
| 1 | Records were rejected but fewer than the threshold for failure |

## Start EP Services

**Mid-Tier**: Primary and Secondary

**Purpose**: Restarts IIS and starts EP Mid-Tier Services for a single Mid Tier. Services should be started on the secondary mid-tiers first and the primary mid-tier last. The mid-tier start order is an EP Product suggestion rather than an EP Batch Facility requirement.

**Usage**: StartEPServices

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



## Stop EP Services

**Mid-Tier**: Primary and Secondary

**Purpose**: Stops EP Mid-Tier Services for a single Mid Tier.

**Usage**: StopEPServices

**Arguments**:

|  |  |
| --- | --- |
| **Name** | **Description** |
| N/A | N/A |

**Return** **Codes**:

|  |  |
| --- | --- |
| **Error Code** | **Description** |
| 0 | Success |
| Non zero | Failure |

**Workflow:**



# EP Batch Service Section 6

The EP Batch Service is a windows service that runs on each EP Mid-Tier Server. This service listens on port 7777 waiting for connection requests form the EP Batch Client. Each request sent from an EP Batch Client is matched to one of the Batch Scripts documented in Section 5 of this document. Request to Batch Script mappings and the file system location of the Batch Scripts are stored in the Applications Settings.

One Batch Script can be run at a time. While the EP Batch Service is running a Batch Script, incoming EP Batch Client requests are queued and the EP Batch Client enters wait mode.

Messages are logged to the Windows Application Event Log. Request types, login ids of requestors and date/times are the kind of details typically logged.

The EP Batch Service and Client are written against the .NET 4.0 Framework and require the downloadable installation found at…

<http://www.microsoft.com/en-us/download/details.aspx?id=17718>

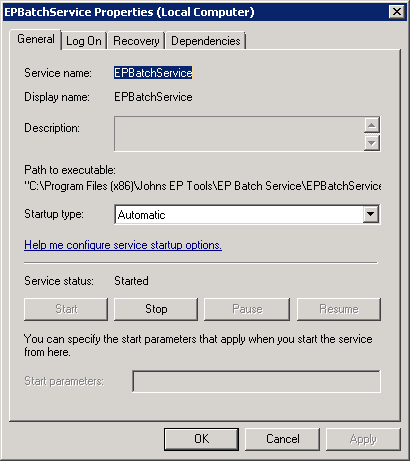
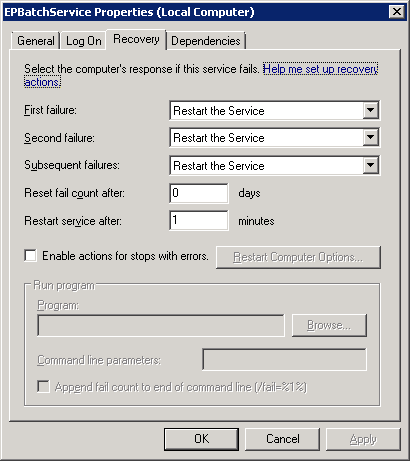
***Note: The EP Batch Client/Server is not intended for production use but rather a tool for the building and testing of an EP Implementation.***

## EP Batch Client/Server Workflow



## Suggested Batch Service Settings

The following settings are suggested using the Microsoft Services application found in Administrative Tools.

## Sample Application Settings

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

<sectionGroup name="applicationSettings" type="System.Configuration.ApplicationSettingsGroup, System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" >

<section name="EPBatchService.Properties.Settings" type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />

</sectionGroup>

<sectionGroup name="userSettings" type="System.Configuration.UserSettingsGroup, System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" >

<section name="EPBatchService.Properties.Settings" type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" allowExeDefinition="MachineToLocalUser" requirePermission="false" />

</sectionGroup>

</configSections>

<applicationSettings>

<EPBatchService.Properties.Settings>

<setting name="RequestType" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>FDE\_POST\_FACT\_DATA</string>

<string>FDE\_STAGE\_FACT\_DATA</string>

<string>POST\_ACTUALS\_DATA</string>

<string>POST\_PLAN\_DATA</string>

<string>REBUILD\_CONTEXTS</string>

<string>RUN\_JI\_MEMBER\_DATA</string>

<string>RUN\_JI\_META\_DATA</string>

<string>START\_EP\_SERVICES</string>

<string>STOP\_EP\_SERVICES</string>

<string>SYNC\_PLAN\_CUBES</string>

<string>SYNC\_ACTUALS\_CUBES</string>

<string>SYNC\_ALL\_CUBES</string>

<string>SYNC\_FOUNDATION\_AND\_STRUCTURE</string>

</ArrayOfString>

</value>

</setting>

<setting name="RequestCommand" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>FDEDataUpdate.bat</string>

<string>FDEStageFactData.bat</string>

<string>FoundationActualsDataUpdate.bat</string>

<string>FoundationPlanDataUpdate.bat</string>

<string>RebuildContexts.bat</string>

<string>JIMemberDataLoad.bat</string>

<string>JIMetaDataLoad.bat</string>

<string>StartEPServices.bat</string>

<string>StopEPServices.bat</string>

<string>PlanCubeSync.bat</string>

<string>ActualsCubeSync.bat</string>

<string>QuartzCubeSync.bat</string>

<string>FoundationStructureUpdate.bat</string>

</ArrayOfString>

</value>

</setting>

<setting name="CommandDirectory" serializeAs="String">

<value>D:\EP\_Batch\</value>

</setting>

</EPBatchService.Properties.Settings>

</applicationSettings>

<userSettings>

<EPBatchService.Properties.Settings>

<setting name="AdvancedRequestTypes" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>POST\_ACTUALS\_DATA</string>

<string>REBUILD\_CONTEXTS</string>

<string>RUN\_JI\_MEMBER\_DATA</string>

<string>RUN\_JI\_META\_DATA</string>

<string>START\_EP\_SERVICES</string>

<string>STOP\_EP\_SERVICES</string>

<string>SYNC\_ACTUALS\_CUBES</string>

<string>SYNC\_ALL\_CUBES</string>

<string>SYNC\_FOUNDATION\_AND\_STRUCTURE</string>

</ArrayOfString>

</value>

</setting>

</EPBatchService.Properties.Settings>

</userSettings>

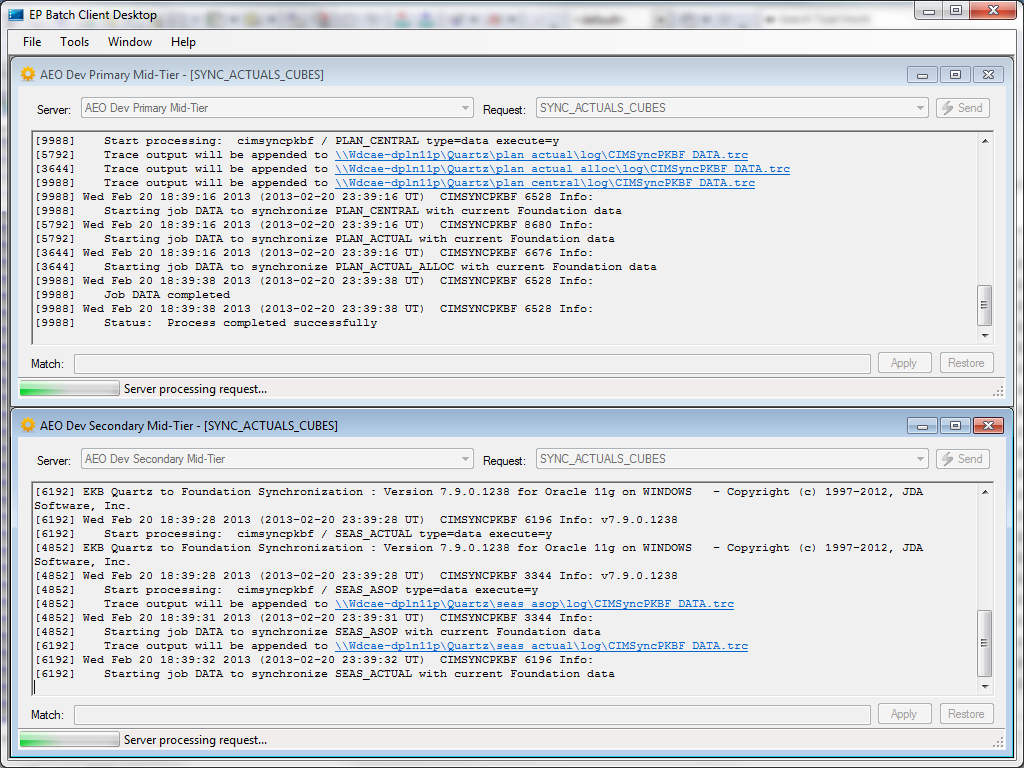
</configuration>

# EP Batch Client Section 7

The EP Batch Client is used to run batch scripts on an EP Mid-Tier system through a TCP connection to the EP Batch Service.

## EP Batch Client Desktop

The EP Batch Client Desktop contains multiple EP Batch Client Session windows and contains the main menu for the EP Batch Client application.



## Menu Items

|  |  |  |
| --- | --- | --- |
| **Menu** | **Menu Item** | **Desciption** |
| File | New… | Creates a new EP Batch Client Session Window |
| File | Exit | Terminates the EP Batch Client. Batch processes in progress will continue to run to completion on the Server. |
| Tools | Advanced Requests | When selected, the Request drop down list will contain all available Server Requests. When not selected, the Request drop down list will contain only items for posting/synchronizing User Plan data. The selection value persists in the User Settings. The Request lists are stored in the Applications Settings. |
| Tools | Options… | Renders the Options Panel. |
| Window | Cascade | Makes all the open EP Batch Client Session Windows the same size and arranges them in a cascading orientation. |
| Window | Tile Horizontal | Makes all the open EP Batch Client Session Windows the same size and arranges them in a horizontally tiled orientation. |
| Window | Tile Vertical | Makes all the open EP Batch Client Session Windows the same size and arranges them in a vertically tiled orientation. |
| Window | Arrange Icons | Cleans up the arrangement of any miniaturized EP Batch Client Session Windows. |
| Window | Close All | Closes all EP Batch Client Session Windows. |
| Help | About EP Batch Client… | Displays the EP Batch Client Info Panel |

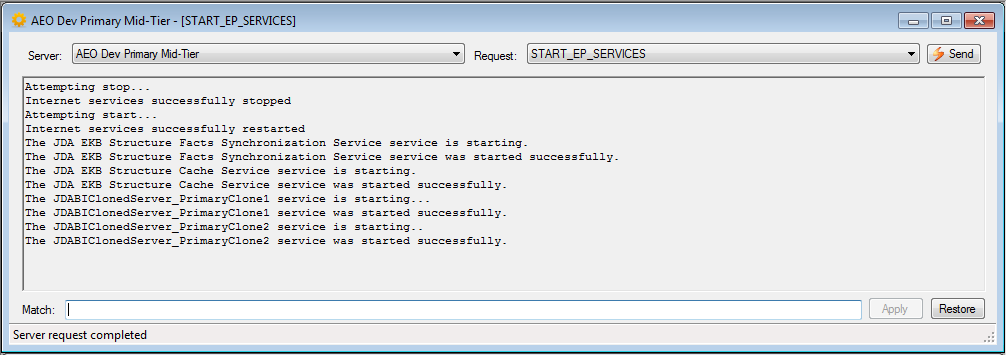
## EP Batch Client Session

The EP Batch Client Session allows the selection of an EP Mid-Tier Server and a Batch Request to run on the server. The send button establishes a connection to the EP Batch Service, sends the Request to the server and displays the Response, real time, from a background thread. The connection to the EP Batch Service is closed upon completion of the Request.

Available Batch Requests include:

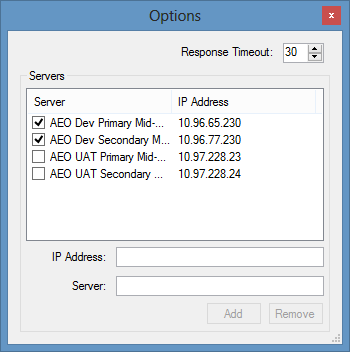
|  |  |  |
| --- | --- | --- |
| **Request** | **Batch Service Action** | **Advanced Request** |
| FDE\_POST\_FACT\_DATA | [FDE Data Update](#_FDE_Data_Update) |  |
| FDE\_STAGE\_FACT\_DATA | [FDE Stage Fact Data](#_FDE_Stage_Fact) |  |
| POST\_ACTUALS\_DATA | [Foundation Actuals Data Update](#_Foundation_Actuals_Data) | x |
| POST\_PLAN\_DATA | [Foundation Plan Data Update](#_Foundation_Plan_Data) |  |
| REBUILD\_CONTEXTS | [Rebuild Contexts](#_Rebuild_Contexts) | x |
| RUN\_JI\_MEMBER\_DATA | [JI Member Data Load](#_JI_Member_Data) | x |
| RUN\_JI\_META\_DATA | [JI Meta Data Load](#_JI_Meta_Data) | x |
| START\_EP\_SERVICES | [Start EP Services](#_Start_EP_Services) | x |
| STOP\_EP\_SERVICES | [Stop EP Services](#_Stop_EP_Services) | x |
| SYNC\_PLAN\_CUBES | [Plan Cube Sync](#_Plan_Cube_Sync) |  |
| SYNC\_ACTUALS\_CUBES | [Actuals Cube Sync](#_Actuals_Cube_Sync) | x |
| SYNC\_ALL\_CUBES | [Quartz Cube Sync](#_Quartz_Cube_Sync) | x |
| SYNC\_FOUNDATION\_AND\_STRUCTURE | [Foundation Structure Sync](#_Foundation_Structure_Update) | x |

Match criteria allows the Response text to be filtered to only contain lines that contain a Match string. The Apply button performs the filtering and the Restore button redisplays the full Response text.



## Options Panel

The Options panel is where EP Mid-Tier Servers are defined.  The checkbox denotes a Server that will be auto opened at application launch. The values in this panel persist in the User Settings.



## Sample Application Settings

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

<sectionGroup name="userSettings" type="System.Configuration.UserSettingsGroup, System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" >

<section name="EPBatchClient.Properties.Settings" type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" allowExeDefinition="MachineToLocalUser" requirePermission="false" />

</sectionGroup>

</configSections>

<userSettings>

<EPBatchClient.Properties.Settings>

<setting name="ServerIpAddresses" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>10.96.65.230</string>

<string>10.96.77.230</string>

</ArrayOfString>

</value>

</setting>

<setting name="ServerNames" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>AEO Dev Primary Mid-Tier</string>

<string>AEO Dev Secondary Mid-Tier</string>

</ArrayOfString>

</value>

</setting>

<setting name="AdvancedRequests" serializeAs="String">

<value>False</value>

</setting>

<setting name="AutoLaunch" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>AEO Dev Primary Mid-Tier</string>

<string>AEO Dev Secondary Mid-Tier</string>

</ArrayOfString>

</value>

</setting>

</EPBatchClient.Properties.Settings>

</userSettings>

</configuration>

## Sample User Settings

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<userSettings>

<EPBatchClient.Properties.Settings>

<setting name="ServerIpAddresses" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>10.96.65.230</string>

<string>10.96.77.230</string>

</ArrayOfString>

</value>

</setting>

<setting name="ServerNames" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>AEO Dev Primary Mid-Tier</string>

<string>AEO Dev Secondary Mid-Tier</string>

</ArrayOfString>

</value>

</setting>

<setting name="AdvancedRequests" serializeAs="String">

<value>True</value>

</setting>

<setting name="AutoLaunch" serializeAs="Xml">

<value>

<ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<string>AEO Dev Primary Mid-Tier</string>

<string>AEO Dev Secondary Mid-Tier</string>

</ArrayOfString>

</value>

</setting>

</EPBatchClient.Properties.Settings>

</userSettings>

</configuration>

# EP Batch Facility Installation Section 8

## EP Batch Client and EP Batch Service

Both the EP Batch Client and EP Batch Service are installed using individual Microsoft Installer packages. By default both of these installer packages install into the C:\Program Files (x86) directory.

The EP Batch Client install package will place a short cut to the EP Batch Client on the users desktop as well as on the Start menu.

The EP Batch Service install package also installs two scripts for the convenience of registering and unregistering the EP Batch Service. The two scripts are:

* **InstallEPBatchService.bat** – This script must be manually run after the install package has been installed. The EP Batch Serivce will be registered with Microsoft Windows, configured (See [Suggested Batch Service Settings](#_Suggested_Batch_Service)) and then started. Be prepared to provide Administrator credentials when prompted.
* **UninstallEPBatchService.bat** – Stops the EP Batch Service and unregisters it with Microsoft Windows.

By default, the EP Batch Service expects the batch working directory (location of scripts and console applications) to be the D:\EP\_Batch directory. This can be changed from the application settings file (See [Sample Application Settings](#_Sample_Applications_Settings)).

## Batch Command Console Applications and Scripts

The batch scripts and console applications are available as a single.zip file:

* **EP Batch Facility [version].zip**

Once the batch scripts have been extracted, the following files need to be manually edited to conform to the current EP Implementation:

* **ActualsCubeSync.bat** – Modify to reflect current Actuals Quartz Cubes (primary and secondary)
* **EPBatchFacility.xml** – Modify to reflect current JI schema password
* **FoundationActualsDataUpdate.bat** – Modify to reflect current Actuals Fact Table Families (primary only)
* **FoundationPlanDataUpdate.bat** – Modify to reflect current User Plan Fact Table Families (primary only)
* **GatherSchemaStats.bat** – Modify to reflect ekbf, ext and Quartz Cube schemas (primary only)
* **GlobalVariables.bat** – Modify to reflect current EP Implmentation (primary and secondary)
* **PlanCubeSync.bat** – Modify to reflect current User Plan Quartz Cubes (primary and secondary)
* **QuartzCubeSync.bat** – Modify to reflect current Quartz Cubes (primary and secondary)
* **QuartzPurgeAgedData.bat** –Modify to reflect current Quartz Cubes (primary and secondary)
* **sftp\_support\_global\_variables.vbs** – Integration configuration parameters that need to be set

For Mulesoft implementations, the following files contain configuration parameters that need to be set…

* **SetEKBSchemaProperties.json**
* **SetEmailProperties.json**
* **SetEXTSchemaProperties.json**
* **SetFrameworkProperties.json**
* **SetHTTPProperties.json**
* **SetJIProperties.json**

**Note**: The ActualsCubeSync.bat, PlanCubeSync.bat and QuartzPurgeAgedData.bat are not the same between primary and secondary. These scripts need to reference only those Cubes whose working directories (temp, log and sort) reside on the particular server. GlobalVariables.bat is different between primary and secondary since this file identifies whether the server is a primary or secondary (See PRISEC in [Global Variables](#_Global_Variables)).

1. As of EP release 8.0, EKBS, EKBP and EKBF have been merged into a single schema. [↑](#footnote-ref-2)