Oracle® Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service

Implementation Guide

Release 18.0

F13381-03

February 2020

ORACLE

Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service Implementation Guide, Release 18.0

F13381-03

Primary Author: Judith Meskill

Contributing Author: Scott Coulter

derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

2 In-Context Help

Send Us Your Comments

Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service Implementation Guide, Release 18.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at http://www.oracle.com.

Preface

Oracle Retail Implementation Guides provide detailed information useful for implementing and configuring the application. It helps you to understand the behind-the-scenes processing of the application.

Audience

This document is intended for administrators and system implementers of Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service documentation sets:

- *Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service Configuration Guide*
- *Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service Release Notes*
- *Oracle Retail Assortment & Item Planning Enterprise Edition Cloud Service Release Notes*
- *Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service Starter Kit*
- *Oracle Retail Assortment & Item Planning Enterprise Edition Cloud Service Starter Kit*
- Oracle Retail Predictive Application Server Cloud Edition documentation set

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

Convention	Meaning
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Implementation

Oracle Retail Merchandise Financial Planning Enterprise Edition Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service act as a platform to create tailored solutions or migrate existing on-premise solutions into the cloud. This guide addresses the process of preparing a custom solution for use in either of these Cloud Service environments.

Because Oracle Retail Cloud Service applications do not support any back-end server access, implementation for an MFP EE CS or A&IP EE CS application is different from an RPAS on-premise implementation. The applications provide online tools to cover all the necessary facets of an RPASCE application roll-out and administration. This includes:

- Building and patching domains from your custom configuration
- Defining nightly, weekly, or ad hoc batch process sequences
- Scheduling recurring batch processes

Required Skills

Since the implementations are based on a retailer- or implementer-provided configuration, working knowledge of the RPASCE configuration tools is essential. The RPASCE configuration tools are supported for offline use on a Windows 7 or 10 system. They are available in the MFP EE CS Starter Kit and A&IP EE CS Starter Kit, and their use is detailed in the *Oracle Retail Predictive Application Server Cloud Edition Configuration Tools User Guide*.

In addition to supplying an RPASCE configuration, the implementer must also prepare the retailer to provide RPASCE hierarchy and measure data load files, as well as to take RPASCE exported measure data files for any downstream integration needs. While the implementer does not call the RPASCE loadHier, loadMeasure, or exportMeasure utilities directly, knowledge of their usage gained from the RPASCE Administration Guide is helpful.

Data files for loading into the applications and exported files for integration with other systems are sent and received from the RPASCE cloud environment via an SFTP site. Knowledge of the use of SFTP software, including an ability to automate such uploads and downloads, is a necessary prerequisite for routine nightly or weekly batch processing jobs.

Batch Framework

MFP EE CS and A&IP EE CS operations require that the administrative user, who will not have command-line server access, must be able to select, initiate, and schedule RPASCE batch activities.

The RPASCE platform includes an Online Administration Tool (OAT) capability, which allows simple parameterization and scheduling of pre-configured batch tasks. The MFP EE CS and

A&IP EE CS introduce an enhancement to the OAT framework that allows a sequence of several batch tasks to be defined. This sequence is built from a list of available batch services, such as measure loading, calculation, segment workspace refresh, and so on. These service tasks run in a defined order, so that you can know, for example, that your daily data updates have been loaded before your workspace refresh tasks are run. The batch tasks are configured to run under the existing OAT framework, so that scheduling them to run once, or on a repeating basis, is the same as for other OAT tasks.

The batch task sequences are defined in a small set of text files, which are specified below, with some examples.

Batch Processes Under the Control of the Implementer

Using the RPAS EE CS batch execution framework, the following are under implementer control:

- List of batch operations to be run, with available parameterization
- Order in which batch operations are to be run
- Scheduling of one or more recurring batch tasks (can be modified by administrator, as needed)

Batch Processes Not Under the Control of the Implementer

Due to the operational and security constraints of the Cloud Service environment, the following are not under implementer control:

- Parallelization: The applications automatically parallelize any applicable batch tasks with a number of processes set to match the provisioned server environment.
- Script file names, file and directory locations: Custom scripting is not supported for this environment, and no knowledge of file system names or locations is necessary in defining and parameterizing the batch task files.
- Incoming and outgoing file (ftp) location: These details are fixed within the RPASCE Cloud Service environment.

Batch Framework Service Catalog

This section describes the batch service.

Batch Exec Service

The Batch Exec service is the controller for all the other services, specifying groups of tasks to be run, their sequences, and top-level parameters.

The Batch Exec service groups are specified in a text file batch_exec_list.txt. In this file, each active line takes this form:

```
batch_type | service | service parameter
```

The first column is an identifier, which may be repeated on several lines to define a grouping of tasks to be run together. The second column indicates which task from the catalogue is being requested. The third column gives parameter details for that task (as necessary). Comments may be placed in the batch_exec_list.txt file by starting a comment line with the hash sign (#).

Here is a sample batch_exec_list.txt file for reference:

```
# Daily Batch Cyle
```

```
daily | waittrigger | daily_upd.txt~ftp~3600
daily | unpack      | daily_upd.tar.gz
daily | calc        | exp_set
daily | measexport  | daily_exp_set
daily | measload    | load_oo_list
daily | sendtrigger | batch_load_complete.txt~ftp
daily | calc        | batch_oo

# Batch Cycle to Load OO
load_oo | measload | load_oo_list
load_oo | calc     | batch_oo

# Weekly Batch Cyle
weekly | calc       | exp_calc_set
weekly | measexport | weekly_exp_set
weekly | hierload   | clnd~14~N
weekly | hierload   | prod~14~N
weekly | hierload   | loc~14~N
weekly | measload   | load_act_list
weekly | measload   | load_oo_list
weekly | calc       | batch_week
weekly | calc       | batch_fcst
weekly | refresh    | refresh_weekly
weekly | autobuild  |
```

In this sample file, three batch task groups are specified: daily, load_oo, and weekly. Note that these names are implementer-defined identifiers; there is nothing special about the names "daily" or "weekly". Each identifier is thus associated with a sequence of tasks, which will run in the order they are listed in the file.

Note also that no information is provided about times or schedules on which these task groups should be run. This information is specified in the RPASCE Online Administration Tool.

The services listed for each batch task group are run in the order specified when that type of batch run is requested through the OAT interface. Details on the individual batch services and what their service parameters mean are described in the following sections.

Load Measures: measload

The Load Measures service allows the loading of one or several measures, the data for which may be found in one or several files. The service can be configured to copy the required data files either from the incoming SFTP location or the in-cloud integration file area. The service will optionally either validate the presence of all data files and treat this as an error condition, or treat the presence of files as optional and continue with no error if the files are not present.

Groups of measures to be loaded are specified in a control file, batch_loadmeas_list.txt, with the columns as follows:

- Load set name
- Parameter type, which must be one of the following:
  - M - Measure names. Nne or more of these lines may be specified.
  - H - Shared measure file, only allowed if the domain is RDM-enabled
  - V - Validate option, if present indicates missing data files are to be treated as an error condition. No third-column parameter is required.
  - C - Copy data file source. Keywords: cloud, ftp; defaults to ftp if not specified.

  - S - Load the measure serially. Current batch framework loads the measure in parallel If this S parameter type is used, the framework will load the measure in succession. Use this option for loading translation measures that cannot be loaded in parallel to a domain.
  - R - Rejected record threshold (optional). Requires third column parameter; it must be a position integer. When this parameter is presented, the process will be stopped (error code 37) with a failure trigger when the number of rejected records exceeds the number specified. If this option is not specified, a warning trigger will be created if there are any rejected records in the measure load; however, the process will continue.
- Parameter value. Relative to the parameter type specified above.

The file name from which each measure loads is specified in the domain configuration, so no file names are given in the batch config files. A domain configuration may register each measure in a separate file, or it may have several measures registered with the same file name.

When loading several measures in a particular batch load set, if the measures are registered with different file names, they must each be listed on their own "M" batch config line. If two or more measures are registered with the same file name, then they must be listed comma-separated on a single "M" line.

Here is an example control file for the Load Measure service:

```
# load on-order measures
load_oo|C|ftp
load_oo|V|
load_oo|R|200
load_oo|M|drtyoou
load_oo|M|drtyoor,drtyooc
```

In this example, data files are to be copied from the incoming SFTP location, and if any files for the listed measures are absent, an error condition will be reported. The measure drtyoou is found in its own file, whereas the measures drtyoor and drtyooc are both to be loaded from the same file. Note that the properties for those measures will need to have been set with the same file name in the domain configuration in order for this feature to be used.

The Validate option checks for required data files in the <domain>/input directory, as well as either the FTP or Cloud data location (if the C parameter is given). This allows the measload task's Validate option to correctly detect files that were previously placed in the <domain>/input location by an unpack batch task.

When measure data files are loaded, some lines in the file may be rejected. This is possibly due to an incorrectly formatted input file or a position that does not exist in the dimension. The RPAS measure load process does not, by default, treat these rejected lines as errors and will continue loading any valid lines from the rest of the file. In order to detect when rejected lines were encountered, since the batch framework does not report this as an error, the loadmeas batch task writes the rejected records count into its own log file and also creates a rejected records warning file in the outgoing FTP area.

The warning file has no contents, providing all relevant information in the file name itself. The file name indicates the name of the measure, the count of rejected records, and a timestamp to indicate when the task was run.

In the example below, the measure apcpfcstslsu had four rejected records when it was loaded on 26-April-2018 at 7:52am:

warning.eebatch_loadmeas.apcpfcstslsu.rejected.4.20180426075212

If the optional |R| parameter is given in the control file, the numerical value indicates a limit to the number of rejected lines, above which the rejections will be reported as an error, rather than a warning. For example, in the load_oo config shown above, the limit is given as 200. If, while

loading any particular measure in this load group, more than 200 rejected record lines are detected, then the task will halt, reporting an error, and the batch sequence that includes this task will also halt. In this way, if a badly formatted or corrupted data file was uploaded, then later batch steps such as calculations or workbook refreshes will be performed.

Export Measures: measexport

The Export Measures service allows the flat-file export of one or more measures, using a control file, batch_export_list.txt, to group (and parameterize) the measure lists for particular export operations. The control file allows multiple parameters to specify the details of each export group.

Here are the columns in the export control file:

- Export Set Name
- Parameter type, which must be one of the following:
  - M - Measure name, format name (optional), and output file name (optional). Separated by |. The format name is the name of the format directive specified in the measure_format_list.txt file. The output file name option is only supported when the control set is using the I parameter. If the I parameter is not specified, the output file name will be determined by the O parameter.
  - F - Filter mask measure
  - X - Base intersection. F or X is required.
  - O - Output file name (optional). This parameter is ignored if the I parameter is specified. One output file for all exported measures. If multiple O parameters are provided, only the first one will be used.
  - I - Flag to use an individual output file for each measure (optional). The default file name is measure_name.csv.ovr unless it is overridden by the output file name option of the parameter of M.
  - S - File share destination. Keywords: ftp, cloud:<app>, where <app> is one of: ri, mfp, rdf, ap, rms. (For the rare case where multiple instances of a single RPASCE application are to be deployed, the second instance may be integrated by using the values mfp2, rdf2, or ap2.) This sends the output file to the indicated Oracle RGBU Cloud Service application, if configured for your environment.
  - C - Compress output. Optional; if single file is output, compress as .gz, if multiple files, compress as .tar.gz.
  - D - Delimiter. Optional character to use in place of comma; to select the | character as the delimiter, specify the keyword "PIPE".

    Note: D simply replaces all commas with the delimiter. It does not work well with string measure values that include commas.

  - U - Uppercase position name (optional). Does not have third column option. If it is specified, the position name in the output file will be converted to an uppercase name.

    P - useDate parameter (optional). Requires either the value "start" or "end" for the third column. It is used for a measure that has a high level on the CLND hierarchy (for example, Mnth). When specified, it will replace the measure CLND level position of each data record with the lowest CLND level (for example, Day) position corresponding to that higher level position. The values start or end are used to

determine whether the starting day position or the ending day position of the corresponding mnth period will be output.

  - N - Specify when to skip NA values in export (optional). Valid options for the third column are "never", "allna", and "anyna".
    * never - Export the corresponded data point even though the measure's values are all NA values. This option essentially exports all data points in the logical space.
    * allna - Do not output the corresponded data point if all measure values are NA values (default mode).
    * anyna - Do not output the corresponded data point if any one measure value is an NA value.

- Parameter value. Relative to the parameter type selected above.

Here is an example control file for the Export Measure service:

```
# Export PoC Plan CP
lpcp|F|lpcpexportb
lpcp|S|ftp
lpcp|M|lpcpbopc
lpcp|M|lpcpbopr
lpcp|M|lpcpbopu
lpcp|M|lpcpeopc
lpcp|M|lpcpeopr
```

For the lpcp export group, the implementer has given a Filter Mask measure, has indicated that the file will be published to the SFTP server location, and has given a list of several measures to be included in the output.

Mace Calculation Service: calc

The Calc service, which indicates that mace is to be run, uses a control file called batch_calc_list.txt. The format of this file is as follows:

```
calc_list | [G or L] | [group or expression] | [group name or expr text]
```

The first column provides an identifier for each group of calc instructions. These identifiers are used to select calculations to be run either directly, or as part of a Batch Exec run. (The identifiers can match the Batch Exec identifier, but this is not required.) The second column, which references either the (G)lobal or (L)ocal domain, is maintained for batch control file compatibility with earlier RPAS versions, but is no longer used. All expressions will be executed in the Global domain context.

The third column indicates whether the calculation to be run is a registered rule group or an individual expression. The final column provides either the name of the rule group to be executed or the text of the expression to be run.

As with the other control files, any line starting with # is ignored and can be used to comment or document the file, as needed.

Here is an example file for the calculation service:

```
# Calc Set for Batch Aggregation Weekly
batch_week | G | group | Batch_GB
batch_week | L | group | Batch_AggW
batch_week | L | group | Batch_InvRoll
batch_week | G | expression | LTWPNSlsR = DRTYNSls1R+DRTYNSls2R
batch_week | G | expression | LTWPNSlsU = DRTYSls1U+DRTYSls2U-DRTYRtn1U-DRTYRtn2U

# Calc Set for Generating Forecast
```

```
batch_fcst | G | group | Batch_Fcst_G
batch_fcst | L | group | Batch_Fcst_L
```

Load Hierarchies: hierload

The service for loading the hierarchies is slightly different from the ones previously described, as it does not require a separate control file. Each hierload instruction in the Batch Exec file provides the details for loading one hierarchy; multiple hierarchies may be loaded, each on a separate line.

The parameter column provided in the Batch Exec file contains three values, separated by the ~ character. The values are: hierarchy to be loaded, purgeAge value, and whether User Defined Dimensions (UDD) are included.

The example from the Batch Exec section, above, contains these sample values:

```
weekly | hierload  | clnd~14~N
weekly | hierload  | prod~14~N
weekly | hierload  | loc~14~N
```

This indicates that in the weekly batch execution, the CLND hierarchy with a purgeAge value of 14 days is loaded, and there are no UDDs in the CLND hierarchy. Similar values are provided in the example for the PROD and LOC hierarchies.

For RDM-shared hierarchies that require User Defined Dimension (UDD) roll-up updates, additional data files can be uploaded and these are processed by the hierload task as well. For each UDD, you can provide a comma-separated values (CSV) file, with a filename of the pattern: [udd dimension name].csv.dat.

Lines in the CSV file must contain three columns:

- Source position
- UDD position
- UDD label

Export Hierarchy: hierexport

The Export Hierarchy service allows the flat-file export of one hierarchy using a control file, batch_exporthier_list.txt, to specify available options. The format of the control file is similar to the Export Measure control file, with the exception that only one hierarchy may be specified at a time.

The three columns in the control file are:

- Export Set Name
- Parameter Type, which must be one of these options:
  - H - Hierarchy Name (required)
  - T - Export Type. F - only formal, I - only informal, A [or omit] - export all positions.
  - L - Header Line export. Output file includes header line for dimension and label columns.
  - U - Export positions of User Defined Dimensions. This implies Header Line export mode; cannot be used with "only formal" export type.
  - O - Output File Name. This is optional; defaults to <hier>.dat.
  - C - Compress result file to .gz (optional)

  - S - File export destination. Keywords: ftp, cloud:<app>, where <app> is one of: ri, mfp, rdf, ap, rms. (For the rare case where multiple instances of a single RPASCE application are to be deployed, the second instance may be integrated by using the values mfp2, rdf2, or ap2.) This sends the output file to the indicated Oracle RGBU Cloud Service application, if configured for your environment.

- Parameter Value. If required, by parameter type.

Here is an example control file for the Export Hier service:

```
# Export PROD hierarchy, compressed
prod_export|H|prod
prod_export|T|F
prod_export|O|prod_exp.dat
prod_export|C|
prod_export|S|ftp
```

In this example, the prod_export grouping indicates that only the formal positions in the PROD hierarchy will be written to a compressed file prod_exp.dat.gz and placed in the outgoing SFTP server location.

Wait for Trigger File: waittrigger

For a recurring batch task (such as a nightly or weekly batch), you may schedule the batch to run at a particular time, but you also need to ensure that it will not start processing until the required input files are available. This requirement is supported by the waittrigger task. The trigger file is a temporary file that must be uploaded to the incoming FTP area last, after all other required files are present. Note that this trigger file will be deleted once the waittrigger task sees it, so you must not specify an actual data file as your trigger. For example, if the batch must wait for prod.dat to be present, you must specify a second file name, such as prod_dat_trigger.txt, and the external integration process that sends the latest prod.dat into the Cloud environment must also create prod_dat_trigger.txt after the prod.dat file is available.

By default, the waittrigger task waits for 23 hours for the trigger file to appear before timing out and reporting an error. A shorter timeout may optionally be specified, given in the number of seconds to wait.

The waittrigger task requires only an entry in the batch_exec_list.txt control file; no separate control file is required. As seen in the example above:

```
daily | waittrigger | daily_upd.txt~ftp~3600
```

This example daily batch task waits up to one hour for the file daily_upd.txt to be present in the incoming FTP location. The third column uses the tilde (~) character as a separator and gives two or three parameters:

- the trigger file name. Simple file names only, no paths.
- a location keyword that indicates where the trigger file will be found:
  - ftp: the FTP server "input" directory
  - cloud: the RGBU Cloud data share location, used when multiple RGBU Cloud apps are integrated together
  - input: the current domain's input directory
- (optional) number of seconds to wait before timing out

When the trigger file is configured to be found in the FTP area, it should be placed under the input directory. This will be the same directory location for any associated data files or data file archives.

Send a Trigger File: sendtrigger

In order to notify other processes, either internal or external to the Oracle RGBU Cloud environment, of the progress of a batch task sequence, the sendtrigger task may be configured. This task takes a two parameter values, separated by the tilde (~) character. The first parameter specifies the trigger file name.

The second portion specifies the destination in which the trigger file will be created:

- ftp - writes to the SFTP server for this application
- cloud:<app> - sends the file to Oracle RGBU Cloud Service indicated by <app>, with valid values ri, mfp, rdf, ap, rms. (For the rare case where multiple instances of a single RPASCE application are to be deployed, the second instance may be integrated by using the values mfp2, rdf2, or ap2.)
- input - writes the file into the input directory of the current domain

The sendtrigger task requires only an entry in the batch_exec_list.txt control file; no separate control file is required. Here is an example entry for a sendtrigger task:

```
daily | sendtrigger | batch_load_complete.txt~ftp
```

This control line indicates that the file batch_load_complete.txt will be created in the SFTP area, once batch execution successfully reaches this point in the daily batch sequence.

Note that no automatic clean-up of the trigger file is performed, so other processes that look for the presence of this trigger file must remove it. If a trigger file from the previous batch run is still in place during a subsequent batch run, the file will remain in place and the file's timestamp will be updated.

Extract Input Files from Archive: unpack

Batch tasks such as hierload or measload expect to find their individual .dat or .ovr files in the incoming file areas. For some integration needs, it may be preferable to send these files together in a compressed archive for faster upload and to ensure that all matching files arrive together. This integration scenario is supported by the unpack task. The unpack task may specify files with the following extensions to be found in the input directory of the FTP site and unpacked into the domain input directory: .tar, .gz, .tar.gz, or .zip. The archive must contain only simple file names and not any subdirectory structure, as this structure would then prevent the files from being found in the <domain>/input directory, where later batch tasks expect them.

The daily batch example above contains this usage for the unpack task:

```
daily | unpack      | daily_upd.tar.gz
```

The task specifies that the archive file daily_upd.tar.gz is expected to be in the FTP input directory, and it will be unpacked into the domain's input directory before any subsequent batch tasks are performed.

Transform File Service

The Transform file service is used for simple integration capabilities for file transformations before hierarchy or measure file loads such as splitting a file, renaming file, swapping columns in the files, and so on. It also provides an option to filter file records based on particular data values. It does not call any RPAS utilities but instead uses some pre-defined functions that can be called and controlled by control file setting changes. It provides some powerful integration capabilities in which the user does not need to create any external process to format the files so it can readily fit into the regular batch framework. For example, a source system might send multiple measure data in a single file but the configured RPAS solution expects individual

measures per file. In such cases, users can call this service to split those files. This process can also be used to transform the exported output files into required formats that can be copied to other locations.

The parameters for this service are provided in a control file batch_xform_list.txt. This service can be invoked from batch_exec_list.txt (Batch Exec Control file) as follows:

<batch_set_name> | transform | <transform_set_name>

Here are the columns in the batch transform control file separated by the PIPE symbol ("|") for the different functions that can be used.

- Transform Set Type
- Parameter Type, which must be one of the following

Parameter Type	Value
I	Input File Path and valid parameter values are cloud for cloud share location (files coming in from other Oracle RGBU Cloud Services), ftp_in for SFTP input, ftp_out for SFTP output, dom_in for Domain Input (default), dom_out (Domain Output), temp for internal temporary ftp_dim_in for <SFTP> /rdm_input/dimdata, ftp_fact_in for <SFTP>/rdm_input/factdatadirectory, rdm_dim_in for <RDM_ROOT>/dimdata, rdm_fact_in for <RDM_ROOT>/factdata to transform interim export files.
O	Output File Path, valid parameter values are: dom_in (Domain input directory; this is the default), dom_out (Domain output directory), ftp_in (SFTP input directory, ftp_out (SFTP output directory), rdm_dim_in (PDS dimension data directory), rdm_fact_in (PDS fact data directory), cloud:<app> (sends file to another Oracle RGBU Cloud Service's SFTP input directory; valid values for <app> are: ri, mfp, rdf, ap, rms; for the rare case where multiple instances of a single RPASCE application are deployed, the second instance may be integrated by using the values mfp2, rdf2, or ap2.
D	Field Delimiter (Default: Comma), for pipe use PIPE.
E	Output File Delimiter (Default:Comma), for pipe use PIPE.
F	Input File Name (Required). Can have multiple F entries, in order to merge multiple files, but at least one F entry is required.
X	Transformed File Name and Field Numbers (in order) as Parameter Value. It can also use Linux Cut command format. One entry for each separate file must be created. It can also take one additional argument. If provided, used as headers for the transformed files.
V	Validate for Files to be Present
Q	To Add Quotes; needed if data can contain comma and input delim is not comma.
L	Filter file based on where filter column and filter value as parameters. By default, it equates the value; however, to use filter value as not equal to, use additional parameter as 'N' after filter value.
U	Create Unique record output files.
C	Copy a column to end of file (copy column number).
W	Swap column from input file (column numbers to swap).
T	Sum column based on key columns (key columns and sum columns).
S	Sort file columns based on key columns.
A	Add a constant value at end of file.
J	Join two columns using a separator and add to the end of the file.
G	To create a complete trigger output file, if needed at the end.

Parameter Type	Value
Z	To compress output files of particular pattern as zip file. It requires the following four parameters delimited by '| ■ <output_compressed_filename> ■ <input_files_prefix> ■ <input_files_extn> ■ <delete_compressed_files_flag (Y/N)>
M	Move or copy files of a particular pattern of files from input to output location. If M is used, only files will be moved or copied. It will not do any further processing to those files. It requires two parameters delimited by |. <move_file_pattern> and delete_file_flag (Y/N). If the delete_file_flag is Y, the source is removed and the operation is a true move. If the delete_file_flag is N, it is a copy instead of a move and the source is not removed.
N	To not push output files to cloud:<app> location at the end and still hold the files in a temporary location for further process to compress the files before pushing the file to cloud:<app> location.
H	Add headers to transformed output file if option X is not used to transform the output file. Parameter must be the complete header for that file and is added as the first record for the output file.
Y	To delete the input files from the input directory after the transform; if not, the used input file will not be detected.

- Parameter Values - Relative to the parameter type selected above.

Example 1: To split a single file into multiple files based on column IDs.

```
rms_oo|F|rms_oo.csv.ovr
rms_oo|I|cloud
rms_oo|V|
rms_oo|X|drtyoou.csv.rpl|1,2,3,6
rms_oo|X|drtyooc.csv.rpl|1,2,3,7
rms_oo|X|drtyoor.csv.rpl|1,2,3,8
```

This example shows an input file split into multiple files using the multiple 'X' option based on column numbers. In the above example, the output files are created in the domain input directory.

Example 2: To split a single file into multiple files based on column IDs and also to filter records based on a column value.

```
rms_inv1|F|rms_inv.csv.ovr
rms_inv1|I|cloud
rms_inv1|V|
rms_inv1|L|5|N
rms_inv1|X|drtyeop1u.csv.ovr|1,2,3,6
rms_inv1|X|drtyeop1c.csv.ovr|1,2,3,7
rms_inv1|X|drtyeop1r.csv.ovr|1,2,3,8
```

In this example, the first only records with fifth column value as 'N' in the csv file and then those will split into multiple files.

Example 3: To copy columns and swap columns before writing the output file.

```
rms_curh|F|rms_curr.csv.ovr
rms_curh|I|cloud
rms_curh|C|3
rms_curh|W|2|6
rms_curh|U|
```

```
rms_curh|X|curh.csv.dat|2,3
```

In this example, the original file only contains five columns. The third column is copied to the end of the file as the sixth column due to the use of option 'C'. Then, column '2' and '6' are swapped due to use of option 'W'. Then it writes out column 2,3 after removing duplicates due to use of option 'U'.

Example 4: To add a constant value to a file and to join two columns based on a separator.

```
rms_patt3|F|rms_prod.csv.dat
rms_patt3|I|cloud
rms_patt3|L|22|NA|N
rms_patt3|A|BRAND
rms_patt3|J|34|22|_
rms_patt3|X|drdvprdattt.csv.ovr.3|1,34,35
```

It is necessary to add a constant value 'BRAND' and also concatenate it with another column and export both the columns.

In this example, the original file only contains 33 columns. It is first filtered for records not equal to 'NA' in column 22. Then it adds a constant value 'BRAND' in column 34. Then, columns 34 and 22 are joined, using the separator '_' that is added as column 35. Finally, the newly added columns 34 and 35 are extracted into an output file.

Example 5: The following sample shows the use of 'E' to create different delimited output file and 'Z' option to compress the output file.

```
mfp_exp_ri|F|ri_mpop_plan.dat
mfp_exp_ri|F|ri_mpcp_plan.dat
mfp_exp_ri|I|temp
mfp_exp_ri|V|
mfp_exp_ri|X|W_RTL_PLAN1_PROD1_LC1_T1_FS.dat|4-
mfp_exp_ri|O|cloud:ri
mfp_exp_ri|E|PIPE
mfp_exp_ri|Z|RI_MFP_DATA|W_RTL_PLAN|dat|Y
```

In this example, use of multiple 'F' options merges two output files and creates one output file with only from column 4 delimited by comma. However, the final output file is created with delimiter as PIP' due to use of option 'E'.

In addition, the use of the 'Z' option compresses the output files of pattern 'W_RTL_PLAN*.dat' created at cloud:ri location into a compressed file as 'RI_MFP_DATA.zip' and deletes the generated file after compressing.

Example 6: The following option shows the use of the 'M' option to copy a set of files from one location to another location.

```
copy_dom_in|I|dom_in
copy_dom_in|O|ftp_out
copy_dom_in|M|*.dat|N
```

This example copies all the files of pattern '*.dat' from domain/input location to sftp_out location. Due to use of option 'N' to not delete the input files, it only copies the file. To move the files, option 'Y' should be used.

Convert Informal Positions to Formal: formalize

The formalize service allows the modifying of current informal positions, which were created on a given hierarchy and dimension having Dynamic Position Maintenance, or DPM, enabled, to make them formal positions. One or more files matching the pattern <hier>.formalize[.extension] must be uploaded via SFTP (or present in the Cloud Integration

directory, if exported from another Cloud Service application). This file (or files) specifies which informal positions to formalize. An option is also available to allow the formalization of all current informal positions on a dimension. Parameters are specified via the batch_ formalizepositions_list.txt batch control file.

The batch_formalizepositions_list.txt control file contains multiple lines to specify each formalize task, each with three required columns, as follows:

1. Formalization Set Name
2. Parameter Type, from these values
   - C - Column index (starting from 1) of the dimension position name in the hierarchy file, as generated by exportHier. [required]
   - D - Dimension to formalize [required]
   - H - Hierarchy of the dimension [required]
   - A - Formalize all informal positions on Dim. No value needed. [optional; omit if sending a <hier>.formalize.dat file]
3. Parameter Value. Varies by parameter type.

If the (A)ll option is not provided, then at least one <hier>.formalize[.extension] file is expected to be available in either the incoming SFTP server or the RGBU cloud data share location (or already present in the Domain input directory, although this is unlikely in the cloud usage). If no formalize files are present, the batch task will report an error. File format details are available in the *Oracle Retail Predictive Application Server Cloud Edition Administration Guide*.

Here is an example specifying a formalization task called prod_sku, which operates on positions of the SKU dimension in the PROD hierarchy:

```
prod_sku|H|prod|
prod_sku|D|sku|
prod_sku|C|1|
```

Rename Positions in a Hierarchy: renamepositions

The Rename Positions service enables the renaming of existing positions in a domain hierarchy. The task is configured by specifying the hierarchy on which positions are to be renamed and looks for a file with the name [hier].rn.dat in the incoming FTP location, or the in-cloud data share location if multiple applications are integrated. If the rename data file is not present, the rename task will exit without error so that the following batch sequence steps may continue.

The format of the rename data file is as specified in the *Oracle Retail Predictive Application Server Cloud Edition Administration Guide*.

The rename positions batch task does not require a separate control file, but may be specified as an entry in the batch_exec_list.txt file, for example:

```
weekly | renamepositions | prod
```

This control line indicates that the weekly batch task will look for a rename positions data file for the PROD hierarchy, prod.rn.dat, and will carry out the renamings specified.

Workspace Refresh by Template Name: refresh

The Workspace Refresh service enables the refresh of particular workspaces with current Domain data. This allows the selection of all workspaces built from a particular template, including the ability to match on partial template names.

The batch_refresh_list.txt contains only two columns: a refresh group identifier and a template name pattern to match. Here is an example of this file:

```
refresh_weekly | mt_wb
refresh_weekly | mp_wb
refresh_weekly | lt_wb
refresh_weekly | lp_wb
```

The example contains only one refresh group, with four template pattern names to match. All workspaces in the global domain or any local domains that are built from templates matching those patterns will be refreshed.

Workspace Delete by Template Name: delete

The Workspace Delete service enables the bulk deletion of all workspaces built from a particular template. This service does not require a separate control file but can be fully specified within the batch_exec_list.txt file. All workspaces built from the given template, either at the master domain or the subdomain levels, will be removed. To remove workspaces from several template types, specify one delete task for each template.

Here is an example of an entry in the batch_exec_list.txt file for this task:

```
weekly | delete | AD_POC
```

Run Segment Build Queue: autobuild

The autobuild service is the simplest to configure in the EE CS batch framework, as it requires no parameters to be specified. When the autobuild service is included in a batch task group, the wbbatch utility is run on master and local domains to invoke the -startQueue build option. Any segments that have been previously queued for automatic build will be created by this call. Since no further parameters are needed, you will note in the Batch Exec section above, that there is no third column for the autobuild service line.

Here is an example of an entry in the batch_exec_list.txt file for this task:

```
batch_weekly | autobuild  |
```

Note that nothing is required after the second pipe (|) character.

Initialize Testing Environment: initrpac

This is the first of two batch tasks that work together to provide automated test capabilities. See the full explanation of RPAC test automation capabilities in Automated Testing with RPAC; the specific activities carried out by the batch tasks are described briefly here.

The initrpac task serves two purposes related to setting up the environment to be ready to run your automated test cases. First, it checks for new or updated test collateral files on the FTP server. Note that there are three archives of test collateral files that can be sent: tests.tar.gz, input.tar.gz, and compare.tar.gz; in each case .zip is a supported alternative to .tar.gz. If any of these collateral file archives are present in the FTP area under the rpac subdirectory, then they will be moved into the internal holding area, ready to be used by the next step in the process; if no new files are present, then the previously sent files will continue to be used.

Note: Incrementally adding test collateral files are not supported; previous file sets of each type are removed before unpacking the new archive, so any updated archive must contain all collateral files of that type. This prevents stale test scripts or data files from being left in the testing environment, which could otherwise cause unexpected test failures.

The second task carried out by initrpac is to place the contents of the input.tar.gz (or input.zip) into the <domain>/input directory. This will be used to place any hierarchy load (.dat) or measure load (.ovr, .clr, .rpl) files into position so that subsequent batch tasks may set the domain into a known state, ready for automated tests to run and verify the expected result values.

The initrpac task entry in the batch_exec_list.txt control file does not require any parameters. It would normally be placed as the first entry in a test-enabled alternate version of a daily or weekly batch execution sequence. See full example in Automated Testing with RPAC.

```
rpac_validate | initrpac |
```

Execute Automated Tests: runrpac

This is the second of the two batch tasks that work together to support automated testing capabilities. The runrpac task executes all automated tests in a single test .XML file (see further information about the RPAC automation testing framework in Automated Testing with RPAC). While the preceding initrpac task must only be run once, you may specify as many runrpac tasks as needed to execute all configured automation tests, possibly at several different points in an overall batch execution sequence.

The runrpac task entry in the batch_exec_list.txt control file takes one parameter that combines an identifier for the test, along with the filename of the test .XML file to be executed:

```
rpac_validate | runrpac   | MFPCS_Sample_Test_1~RT01_MT_WB.xml
```

In this case, the test file RT01_MT_WB.xml will be executed under an identifying title "MFPCS_Sample_Test1". See Automated Testing with RPAC for a full example of a test-enabled batch execution sequence.

Summary test results will be visible in the output log for the batch execution (visible in the Online Administration dashboard), and full test result details will be available in the log file archive that is sent to the FTP server after the batch execution completes.

Automated Testing with RPAC

The RPASCE Pluggable Automation Component (RPAC) utility is supported for use with Enterprise Edition deployments. RPAC tests are specified in XML-format text files and cover a range of RPASCE Domain and Segment activities. Note that RPAC does not support the testing of GUI functions and is not a performance testing tool. In order to support the validation of a newly installed or patched environment, in the context of configured daily or weekly batch operations, RPAC for Cloud deployments is supported through new entries in the EE Batch task catalog. These tasks allow a pre-production domain to be set to a known state through a combination of hierarchy load and measure load files, and then can compare both Domain and Segment Workspace measures to known values represented either directly in the test xml files or in data comparison files. This is similar to a measure data load file, but used only for comparison rather than for loading.

Three types of collateral files are involved in the RPAC testing process:

- Input data file set: a group of hierarchy (.dat) and measure (.ovr, .clr, or .rpl) data files that should be loaded into the domain before any RPAC tests are run. Uploaded to FTP site in the rpac directory as input.tar.gz or input.zip.
- Test file set: one or more .xml files where tests and test suites are defined using the available set of RPAC tags and attributes. Uploaded to FTP site in the rpac directory as tests.tar.gz or tests.zip.

- Comparison data file set: an optional way to efficiently validate that one or more measures currently contain an expected set of values. Uploaded to the FTP site in the rpac directory as compare.tar.gz or compare.zip.

Each of these collateral file archives, once sent through the FTP interface, will be kept internally to be used every time an RPAC-enabled batch execution sequence is run. Updates to the collateral files can be sent to the FTP site before the next call of the initrpac batch task and will be brought into the active environment at that time. Note that when any of the collateral file archives is updated, the previous contents are entirely removed from the internal storage area, so the replacement archive file must be a complete set of files of that type. This prevents stale test scripts or data files from being left in the environment.

The two EE Batch tasks, initrpac and runrpac, are detailed in the batch task catalog in "Initialize Testing Environment: initrpac" and "Execute Automated Tests: runrpac". The initrpac task is expected to be run once, at the start of the RPAC-enabled batch exec sequence; the runrpac task can be called multiple times, including at separate points during the batch exec sequence, if needed. Here is an example batch execution sequence that shows how an existing weekly batch specification might be augmented with RPAC tests:

```
# Standard Weekly Batch Cycle
weekly | unpack    | weekly_sales.tar.gz~ftp
weekly | hierload  | prod~14~N
weekly | hierload  | loc~14~N
weekly | measload  | load_oo_list
weekly | calc      | batch_fcst
weekly | autobuild |

# RPAC-enhanced Batch Cyle
validate | initrpac  |
validate | hierload  | prod~14~N
validate | hierload  | loc~14~N
validate | measload  | load_oo_list
validate | runrpac   | RPAC_Domain_Tests~DomainTests.xml
validate | calc      | batch_fcst
validate | runrpac   | RPAC_Segment_Tests~SegmentTests.xml
```

The first section, labeled "weekly", represents a weekly batch sequence that might run at midnight every Saturday. Note that updated hierarchy and measure data files for the week are sent through FTP in an archive file named "weekly_sales.tar.gz" using the unpack task.

The second section shows how the weekly batch sequence has been augmented with RPAC tests and named "validate". Note that the unpack task from the weekly sequence has been left out, and in its place initrpac is called to place the test data input files into the domain. If new or updated RPAC test collateral files have been placed on the FTP server, they will be brought in at this point and used.

There are two sets of RPAC tests in this sequence, specified by the runrpac task entries. The first runs immediately after the hierarchy and measure files are loaded, and validates expected values in the domain. The second test set is executed after some further calculations have been run, and builds one or more segments, then validates values within them as well.

When RPAC-enabled batch sequences are run, the primary log file, which is available through the Online Administration dashboard as well as through the FTP log archive package, will show a brief summary of test results. Full test details and log files are available in the complete log archive package from the batch exec run, available in the FTP area once the execution has completed.

For full details on the contents of an RPAC test .xml file, and all the tags and attributes that are available for specifying RPAC tests, see "Appendix B: RPAS Test Automation" in *Oracle Retail Predictive Application Server Fusion Client Administration Guide*. Note that the latest

version of this guide specifies which RPAC features are available for Cloud deployments. Due to Cloud security constraints, some RPAC features, primarily the <SHELL> tag, have been disabled; however, inclusion of RPAC tests as a step in existing batch execution sequences should fully compensate for this restriction.

Domain Creation

This section describes domain creation.

SFTP Upload Location

Oracle RGBU cloud services include an SFTP site for incoming and outgoing file transfers. When you upload files via SFTP, a signal must be sent to indicate that you are finished uploading files and the files are ready to be transferred internally to be available for use by the application. This signal takes the form of a subdirectory named "COMMAND", containing one file named "COMPLETE" (file contents not used, file may be empty). In each of the directories described below, if the COMMAND subdirectory does not yet exist, you must create it. Then, when finished uploading files to a particular directory, add the COMPLETE file into the COMMAND subdirectory.

For the purposes of building the domain, three subdirectories in the SFTP site are used:

config

For uploading the domain configuration into the cloud environment, create an archive (either .zip or .tar.gz) containing the contents of the config directory (without the top level config folder). This archive file must be named as <config_name>_config.zip or <config_name>_ config.tar.gz. This archive file must be placed in the config subdirectory on the SFTP server. It may be updated as often as necessary in support of domain build or patch activities. Remember to create the COMMAND/COMPLETE trigger when you are finished placing files in this directory.

Example

The ascs_config.zip must contain the following contents:

- ascs folder - this is the folder with ascs application configuration.
- ascsDashboardSettings.json - required for dashboard.
- ascsHelpConfig.json - required for Help.

batch_control

The set of batch process control files, as detailed in the previous section, must be uploaded to the batch_control subdirectory within the incoming SFTP location. These files are placed into the domain environment when the domain is built and can be updated later by running the domain patch task. Remember to create the COMMAND/COMPLETE trigger when you are finished placing files in this directory.

input

The initial domain creation process requires at least the .dat files for all hierarchies specified in the domain configuration. Normally, it is desirable to have an initial set of measure data load files available at domain build time as well. All of these files must be placed in the input directory of the incoming SFTP location. In addition to the domain build and patch processes, any use of the measload or hierload tasks in the batch framework always checks for incoming data files from this directory. Remember to create the COMMAND/COMPLETE trigger when you are finished placing files in this directory.

jse

The Enterprise Edition includes optional support for one or more Java Special Expression (JSE) extension libraries. JSE extensions are encoded into one or more Java .jar files, and the new expressions may then be used in your domain configuration. To include your .jar file(s) in the domain environment, they must be uploaded via SFTP to a subdirectory named "jse". The .jar file(s) must then be named in the Bootstrap OAT parameters, as described in JSE Jar Files.

Note: Remember to send a COMPLETE file into the top-level COMMAND directory once all files for the domain build process have been uploaded as described within SFTP Upload Location.

Bootstrap Environment

A newly provisioned RPAS EE CS cloud environment is set up with a bootstrap configuration that allows the implementer to log into the RPASCE Client and access the Online Administration Tool (OAT) interface before the domain has been built. The bootstrap OAT configuration allows only tasks required to construct a domain. Once the domain has been constructed, both the domain tasks and activities as well as the bootstrap activities will be available. This allows the domain to be re-built from scratch multiple times, should this be required.

OAT Parameters

A few parameters must be specified when initiating a domain build process through OAT. The implementer must supply these values:

Config Name

The name under which the configuration has been saved. For those familiar with the RPASCE domain construction process, this is the name that is internally passed as the -cn parameter to rpasInstall. A drop-down list offers choices based on the available domain config archive files in the incoming FTP area.

Partition Dim

The dimension on which the domain will be partitioned. The domain is constructed with one subdomain for each position in the given dimension. This must be a level of separation that fits with the intended workflow for individual users so that, when possible, most users' daily tasks relate to only one subdomain. This lessens contention when many users are active in the system.

JSE Jar Files

If any Java Special Expression .jar files are used by the domain configuration, they must be indicated here, by .jar file name only. If you are including multiple .jar files, provide a comma-separated list of .jar file names. This field is optional and must be left blank if JSE extensions are not used. If listed, the .jar files named here must be present in the incoming FTP area under the jse directory, as indicated in the section jse, and the domain build process will halt with an error if the named .jar file(s) are not present.

Input Archive

If input files with hierarchy and measure data are being sent in an archive file as described above, the archive file name must be given here. A drop-down list offers the names of any archive files in the FTP input directory area. If individual data files are being sent, this selection may be left blank.

Batch Group

Once a domain has been built successfully, a named group of batch operations may be specified (typically including measure data loads and mace calculations). This operation sequence must be one batch_type entry in the Batch Exec control file, batch_exec_list.txt (described above in Batch Exec service section).

Overwrite

In the case where the domain has already been built once, and the implementer must rebuild the domain from scratch, which might occur because a non-patchable change has been made to the configuration, this option must be selected. If it is left in the default unselected state, then the domain build process will halt and report an error, rather than overwrite the existing domain.

Domain Build

The domain build process automatically carries out the following steps:

1. Basic validation of the given config name and partition dimension.
2. Ensure that a configuration with the given config name has been uploaded.
3. If the overwrite flag is false, ensure that there is no existing domain. It reports error if domain exists.
4. If the overwrite flag is true, remove the existing domain.
5. Build the domain using the config name and the partition dimension as specified in the OAT parameter screen.
6. Copy any users and user groups from the bootstrap domain environment into the domain environment.
7. Copy the uploaded batch control text files into the domain from the SFTP location.
8. Run post-domain-build batch group.
9. Add the domain details into the provisioned RPASCE Client configuration.

Once all these steps have completed successfully, which you may validate using the Online Administration Tool dashboard, then you may run the secondary OAT task that performs a restart on the RPASCE Client. This is required after the first time the domain is built, to enable logging in to the new domain.

2

In-Context Help

This chapter describes how to configure In-Context Help for solutions based on RPASCE. In-Context Help is a resource to access relevant help topics, in the format of html and video, within the application. At present, it focuses on help topics related to the dashboard and the workspace. The InContext Help file is located under config/Help in the RPASCE Client install folder. The naming convention is <solution-name>HelpConfig.json.

Navigating to Help Topics on RPASCE

You can navigate to the help topics in the following ways:

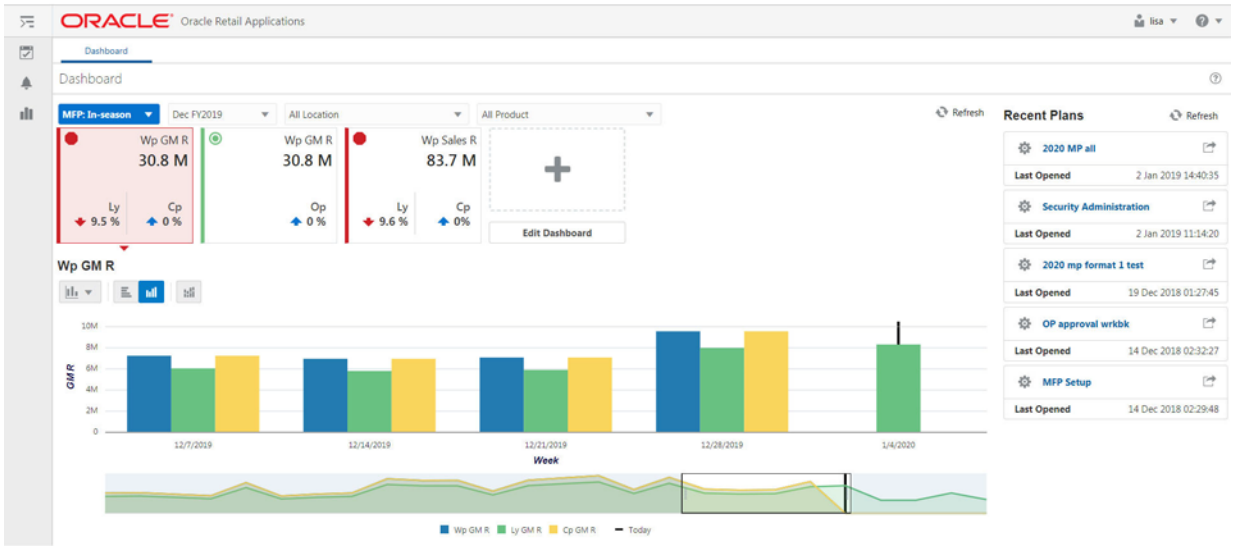
Dashboard

The help topics for the dashboard are added to the following two levels:

- All: The generic topics related to MFP or A&IP are added to this level.
- Report: This consists of topics related to dashboards such as the effective usage, how to analyze the metrics, and so on.

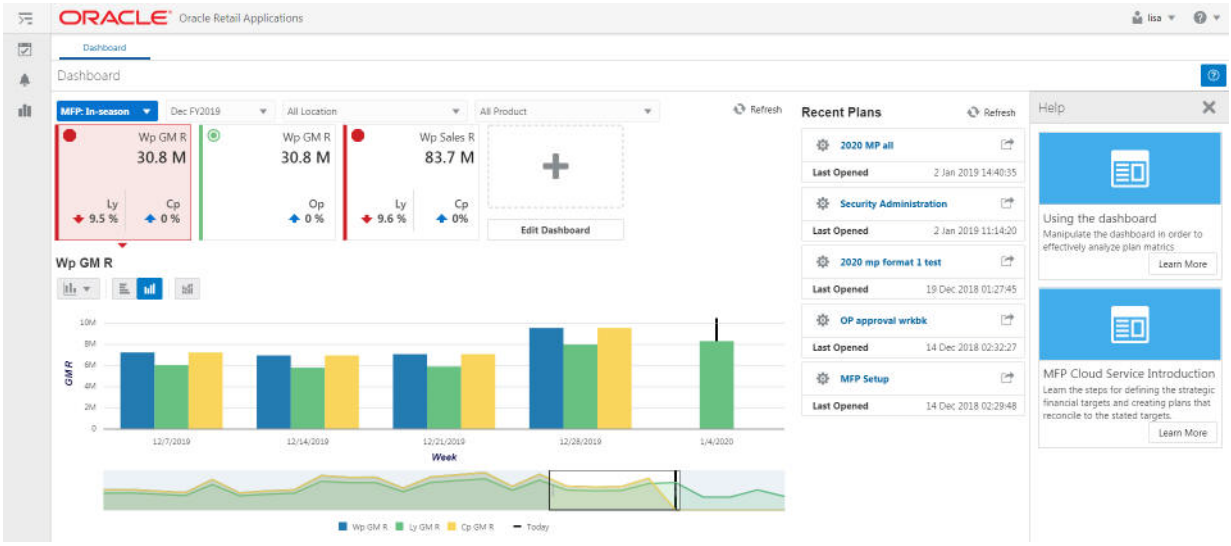
Figure 2–1 shows the view of a dashboard.

Figure 2–1 Dashboard Window



The help topics for the dashboard are visible on the right side panel, as shown in Figure 2–2.

Figure 2–2 Dashboard Help Topics

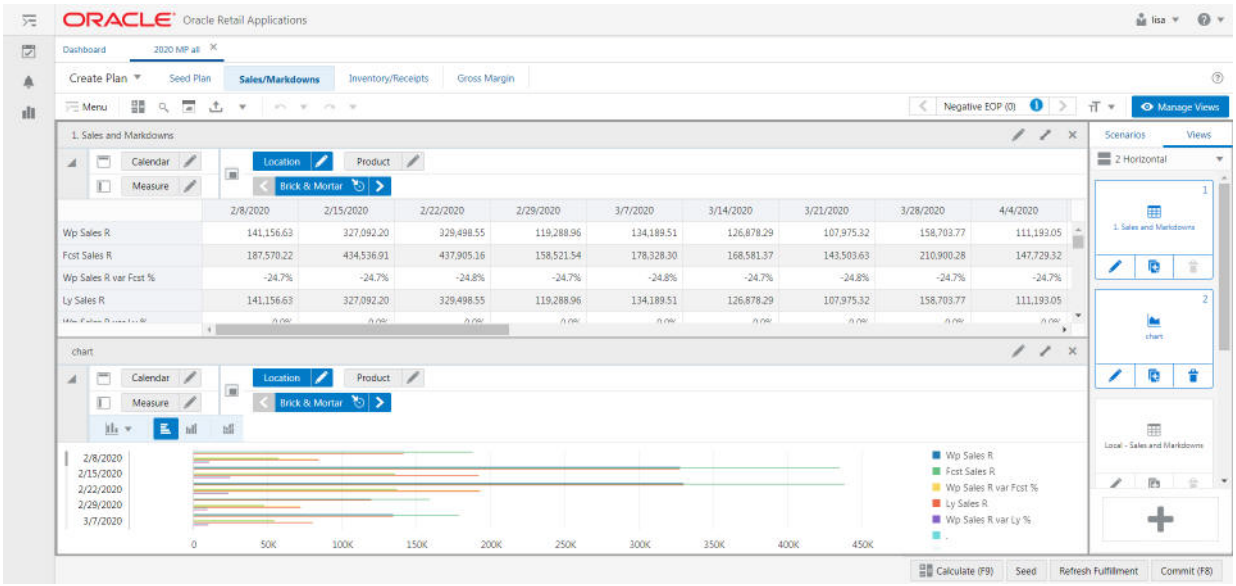


Workspace

The workspace contains the actual content related to MFP or A&IP. Here the topics are aligned with respect to the different levels of the Taskflow.

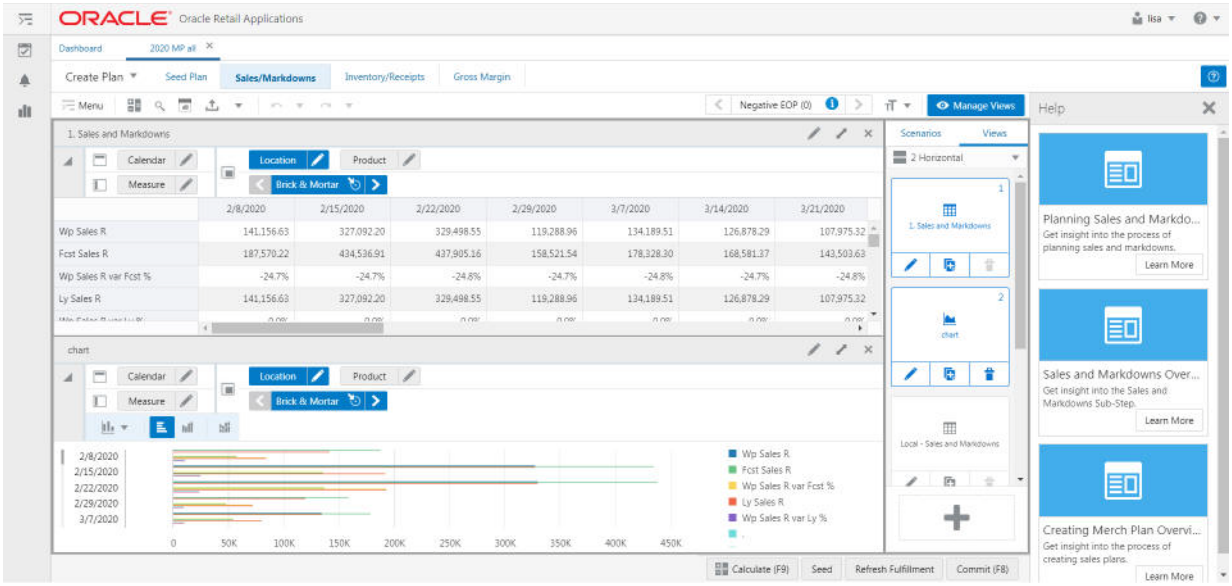
Figure 2–3 illustrates the workspace for the product MFPRCS.

Figure 2–3 MFPRCS Workspace



Here the Step, Tabs, and View are visible.

Figure 2–4 MFPRCS Workspace with Help



Creating the Contextual Help Configuration File

The specifications related to Contextual Help for the RPASCE dashboard and workspace are implemented by creating a configuration file. This file is created outside of the RPASCE Configuration Tools and is deployed in the RPASCE Client application. The contents of this configuration file are used by the RPASCE Client to determine how to organize and display the help topics in the dashboard and the workspace.

Although a Contextual help configuration file can be created from scratch, in most cases, it is simpler to modify an existing version of the file to incorporate any desired changes.

Using JSON in the Contextual Help Configuration File

The contents of the Contextual Help Configuration file are formatted as a JSON (JavaScript Object Notation) object. JSON is a common flexible information encoding notation used frequently in cloud applications; it is more compact and, when properly formatted, more readable than the XML format. (Importantly, it is also not subject to some security concerns that are present when using XML for information encoding.)

JSON is a simple and straightforward format; information about the specifics of the format is readily available online.

Structure of Contextual Help Configuration File

The configuration file is divided into three levels: ALL, REPORTS, and WORKBOOKS. All three levels are of type JSON Object. The ALL level is the generic level. REPORTS and WORKBOOKS are children of level ALL.

Table 2–1 Configuration File Levels

Level	Description
ALL	Describes the generic help topics related to the RPASCE solution in use.
REPORTS	Contains the help topics related to the dashboard.

Table 2–1 (Cont.) Configuration File Levels

Level	Description
WORKBOOKS	Contains the help topics for the workbook and its sub-categories, such as Task, Step, Tab, and View.

The generic JSON structure for any solution is as follows:

```
    "helpTopics" : []
    "reports" : {"helpTopics" : [] }
    "workbooks" : {"helpTopics" : []
}
```

Help Topic Building Block

The help topics object as a whole is a JSON array of collection of attributes. This help topics object is the building block for all the different levels.

The following JSON snippet explains the generic helpTopics object structure:

```
"helpTopics" : [{
"name" : "Help Topic 1",
    "description" : "Description 1",
    "url" : "URL 1",
    "type" : "Type 1",
    "imageSrc" : "Image 1",
    "color" : "Color 1"
},{
    "name" : "Help Topic 2",
    "description" : "Description 2",
    "url" : "URL 2",
    "type" : "Type 2",
    "imageSrc" : "Image 2",
    "color" : "Color 2"
}]
```

Table 2–2 list the help topic properties.

Table 2–2 Help Topic Properties

Property	Value Type	Description
Name	JSON String	The name of the topic.
Description	JSON String	A short description of the topic.
URL	JSON String	The URL link to the help topic.
Type	JSON String	The type of the resource. Values are: document, image. or video.
ImageSrc	JSON String	The path to the image file to be displayed in the Help topic card, for example, an icon representing a video or an illustrative screenshot.
Color	JSON String	The color in which the help topic tile should be visible. Color is displayed at the top of the Help topic card. It is typically used to visually distinguish between help formats (document or video), but may be used for a variety of purposes. Values are: lightblue, red, lightgreen, purple, blue, grey, orange, turquoise, green.

Key Naming Convention

The naming convention of the key depends upon the level or sub-level of each element. Here is an example of the naming conventions at different levels.

Consider the solution in use is mfprcs.

Table 2–3 MFPRCS Key Naming Example

Level/Sub-Level	Key Example	Description
All	NA	No need of the key as help topics are added to the root of the JSON.
Reports	reports	
Reports > Dashboard	reports.dashboard.id	The key must match the name provided for reports in the Taskflow_MultiSolution.xml file.
Workbooks	workbooks	
Activity > Task	mfprcs.Activity1.Task1	The task name must match the entry provided in Taskflow_ MultiSolution.xml file for the specific task.
Activity > Task > Step	mfprcs.Activity1.Task1.Step1	The step name must match the entry provided in Taskflow_ MultiSolution.xml file for the specific step.
Activity > Task > Step > Tab	mfprcs.Activity1.Task1.Step1. Tab1	The tab name must match the entry provided in Taskflow_ MultiSolution.xml file for the specific tab.
Activity > Task > Step > Tab > View	MT_TB01_WS01	The view name must match the entry provided in Taskflow_ MultiSolution.xml file for the specific view. The view key name is unique, as it can be added anywhere under Task, Step, or Tab from the solution.

JSON Structure of Contextual Help Configuration File

Here is an example of JSON object containing all the three levels and the help topics related to each of them. The maxTopics in the following snippet defines how many topics can be visible on RPASCE. This value must be increased if you want to show more help topics at a given level than the value of maxTopics. If, for a specific level, there are fewer than maxTopics topics, it fetches the remaining topics from its parent. In the following snippet the maxTopics for workbooks is set to 2 and overrides the maxTopics for the root, which is set to 3 for the workbooks level. Also, since no maxTopics is set for reports, the maximum topics for this level is capped to 3, which is fetched from the root level.

```
{
  "maxTopics" : "3.0",
  "helpTopics" : [ {
    "name" : "MFP Cloud Service Introduction",
    "description" : "Learn the steps for defining the strategic financial targets
and creating plans that reconcile to the stated targets.",
    "url" : "http://docs.oracle.com/cd/E75764_
01/merchfinplan/pdf/cloud/161/html/retail_implementer_
```

```
guide/output/introduction.htm#introduction",
    "type" : "document",
    "imageSrc" : "",
    "color" : "turquoise"
  } ],
  "reports" : {
    "helpTopics" : [ ],
    "reports.dashboards.id" : {
      "helpTopics" : [ {
        "name" : "Using the dashboard",
        "description" : "Manipulate the dashboard in order to effectively analyze
plan matrics",
        "url" : "http://docs.oracle.com/cd/E75764_
01/merchfinplan/pdf/cloud/161/html/retail_implementer_
guide/output/dashboard.htm#dashboard",
        "type" : "document",
        "imageSrc" : "",
        "color" : "turquoise"
      } ]
    }
  },
  "workbooks" : {
    "maxTopics" : "2.0",
    "helpTopics" : [ ],
    "mfprcs.Activity1.Task1" : {
      "helpTopics" : [ {
        "name" : "Overview of Merch Plan Targets",
        "description" : "Learn about the steps associated with creating and
monitoring targets",
        "url" : "http://docs.oracle.com/cd/E75764_
01/merchfinplan/pdf/cloud/161/html/retail_implementer_
guide/output/CreateMerchPlanTargets.htm#create_merch_plan_targets_task",
        "type" : "document",
        "imageSrc" : "",
        "color" : "turquoise"
      } ]
   }
}
```

Editing the Contextual Help Configuration File

Help topics can be edited or added directly under the levels ALL and REPORTS. For level WORKBOOKS, the implementer can add or edit under Task or can add or edit under a specific sub-level (Step, Tab, or View).

The following examples indicate where the implementer can add help topics at different levels.

- Adding or editing the help topic for level ALL.

  The implementer can add the help topic object directly under the root of the JSON under the property helpTopics. For editing, the implementer must search the name of the help topic in JSON and edit any of the required properties.

- Adding or editing the help topic for level REPORTS.

  Here the implementer must add the help topic under the reports object of the JSON. The implementer must search for the key reports and then add the help topic under the attribute helpTopics. Similarly, any particular help topic can be edited by searching the name of the help topic.

- Adding or editing the help topic for sub-level Step under level WORKBOOKS.

To add a topic under sub-level Step, the implementer must search for the Step key and add the help topic. For editing, the implementer must search for a particular help topic and edit any of the properties as required.

- Adding or editing the help topic for sub-level View under level WORKBOOKS.

  To add a topic under sub-level View, the implementer must search for the View key and add the help topic. For editing, the implementer must search for a particular help topic and edit any of the properties as required.

A

Appendix: Exit Codes

This appendix describes all non-success exit codes from the Batch Framework services and batch administration tasks.

All EE batch scripts have consistent exit codes. Codes from 1 to 22 come from the BSA framework (although only 6 and 13 are commonly used by EE batch and so are included in the table below). Codes of 30 and above are from EE batch scripts themselves and are also listed in Table A–1.

Table A–1 lists the common (non-success) exit codes from the EE batch scripts and the BSA framework.

Table A–1 Common Exit Codes

Code	Reason
6	too few args / missing arg
13	invalid domain path
30	required environment variable is not set
31	batch config file is not found
32	selected batch config entry is not found in file
33	invalid or missing info in batch config file
34	unknown error detected in RPAS utility log output
35	file/directory not found when moving or copying files
36	file/directory permission error when moving or copying files
37	measure load exceeded reject record limit.

Note that in a live OCI-provisioned environment, it is not expected that customers will see any of these error codes except 31 through 33. These codes indicate issues in the customer-provided batch config files.

Table A–2 lists additional exit codes from eebatch_exporthier.ksh, eebatch_exportmeas.ksh, eebatch_loadhier.ksh, and eebatch_loadmeas.ksh, that result from the exit codes of the underlying RPAS binary utilities (exportHier, exportMeasure, loadHier and loadMeasure). The exit codes from the binary utilities are reported by the EE Batch Framework as being 100 more than the raw utility results. This prevents overlap between the BSA/EE script result codes and the RPAS binary utility result codes. If loadHier itself returns an error code of 5, then the EE batch framework will report the error as code 105.

Table A–2 Additional Exit Codes

Script	Code	Reason
eebatch_exporthier.ksh	103	Cannot run exportHier on a subdomain
eebatch_exportmeas.ksh	102	Cannot export HSA measure; generic exportMeasure error
	103	exportMeasure encountered RPAS exception, C++ exception or unknown exceptions
eebatch_loadhier.ksh	101	loadHier purgeAll got Exception
	102	calendar prepending error
	103	RPAS HierarchyException
	104	PartitionException: loadHier cannot add new position to partition dimension
	105	Cannot run loadHier while RDM repartitioning is in progress; Internal error: domain is neither Master nor Simple; other unknown exceptions
	106	Unknown exceptions
	107	Cannot run loadHier during RDM update
eebatch_loadmeas.ksh	98	Internal aggregation error
	99	Internal aggregation error; cannot load in CLR mode because the measure does not have a clear intersection; unknown internal error
	101	RPAS exception or internal error
	102	C++ exception

It is not expected that customers will encounter any of the RPAS exceptions, internal errors, or C++ exceptions, which indicate corrupted data or a programming error.