Oracle Retail® Predictive Application Server Administration Guide Release 11.1.12.1 July 2006

ORACLE

Contents

Customer Support

- https://metalink.oracle.com

When contacting Customer Support, please provide:

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

1
Introduction

Overview

All Foundation-based products require setup and administration activities to be performed.

- Domain administration
- User account management
- User and workbook template administration
- Hierarchy maintenance
- Measure analysis
- Workbook auto build maintenance
- Translation administration

System Administration Workbooks

Using the administration workbooks, designated employees manage other employees' use of the Oracle Retail Predictive Solutions. System administrators use the administration workbooks to perform the following:

- Set up and maintain users and user groups.
- Manage users' access to specific workbook templates and individual measures.
- Modify the labels associated with users and user groups.
- Modify the labels associated with workbook templates and template groups.
- Create custom workbook templates and template groups.
- Edit the contents of translation tables to support multiple-language use of the application.
- Specify the type, frequency, and format of workbooks in the automatic build queue.

Workbook and Wizard Descriptions

- User Account Management wizards, a set of wizards for setting up and maintaining users and groups
- Security Administration Workbook, a workbook for setting up and maintaining user/template, user/measure, and template/measure access rights
- Workbook template definition, a workbook for specifying parameters of a custom workbook template
- Translation Administration Workbook, a workbook for managing the foreign language translation of strings and label text throughout the application
- Workbook auto build maintenance, a workbook for managing the workbook auto build queue

General Workbook Procedures

Change a Workbook's Calculation Method

There are two types of calculation modes that can be set in the RPAS client.

1. "Deferred" calculation mode is the most common and allows you to make multiple edits in a workbook before recalculating the data. In this mode, the edits are effectively "queued" and executed once the user hits "Calculate."
2. "Automatic" calculation mode forces the workbook to be recalculated every time a cell is changed. This forces immediate communication from the worksheet back to the database. In this mode, there may be a pause between one data change and your ability to effect the next change.

For efficiency and usability purposes we recommended that you operate in "deferred" calculation mode.

Set the Workbook to Deferred Calculation Mode

Click the Edit menu and select Manual Calculation.

Return the Workbook to Automatic Calculation Mode

Click the Edit menu and select Automatic Calculation.

Send the Queue of Data Changes to the Server

Click the Edit menu and select Calculate Now.

Refresh and Export Data

Refresh the Data in a Worksheet

The Refresh feature allows you to update a workbook with the data that is currently stored in the domain. This lets you work with the most current data without having to rebuild the workbook.

Refresh the Data for all Worksheets and Measures in the Workbook

From the File menu, select Refresh.

Export the Current Worksheet View to an Output File

1. From the File menu, select Export Sheet. The Save As dialog box is displayed.
2. In the Save In field, select a directory on your computer where you want to save the export file.
3. In the File Name field, type a name for the export file.
4. Click the Save As Type drop-down list and select a file type for the export file.
5. Make a selection for each of the following:
   - **Delimiter** – Specify the character used to separate information in the output file. Standard choices are Tab, Comma, or Space, but you can specify a different delimiter by selecting the Other radio button.
   - **Labels** – Specify the format of the label headers across the top of cells in the output file. The options are:
     - **Do Not Include** – no labels provided
     - **Include Once** – one label placed across the top of each section of related cells
     - **Repeat** – a separate label, repeated as necessary, appears atop each cell
     - **Descriptions** – Specify whether to identify dimensional positions in your output file with concise system names (for example, SKU00012) or the descriptive labels (for example, Cashmere Sweater – L – beige) assigned to each position.
6. Select **Save** to export the file.
7. Click **OK**.

Insert Measures into an Open Worksheet

If you have the necessary access rights, you can insert a new measure or group of measures into a workbook that is already open. This functionality reduces the need to build new workbooks whenever a view of currently unrepresented measures is required. The ability to insert new measures into workbooks that are already open is particularly useful in the context of establishing access to alerts.

Use the following procedure to select a measure or group of measures to be inserted in a currently open workbook.

> **Note:** A worksheet must be open and active for the Insert Measure menu option to be enabled. Any measure(s) selected to be inserted in the workbook will be placed on the currently active worksheet.

1. From the Edit menu, select Insert Measure.
2. From the Metric list box, select the desired metric(s). If a measure exists that matches the combination of attribute types selected in the preceding steps, its name/label will appear in the Measures list box. Multiple measures may appear here if multiple selections were made in any of the preceding steps.

   > **Note:** You can right-click in the Metric window to produce a quick menu, wherein you can toggle the display between metric name (the system name) and metric label (as it appears in the workbook).

3. Select the desired measure(s) in the Measures list box.
4. Click **OK**.

Global Domain

Overview

"Global Domain" is a type of domain structure that provides the ability to view data from multiple domains and to administer common activities of an RPAS domain and solution.

Domains can be built in one of two methods:

1. Simple Domain – this is the traditional, stand-alone domain that has no visibility to other domains.
2. Global Domain – this is a domain environment that contains two or more "local" domains (or "sub-domains") and a "master" domain that has visibility to all local domains that are part of that environment.

There are two primary functional benefits in using a Global Domain environment. The first feature is the ability to have a global view of data in workbooks. You can build workbooks with data from local domains, refresh global workbook data from local domains, save global workbooks, and commit the data from global workbooks to the individual local domains.

"Local" domains are typically organized ("partitioned") along organizational structures that reflect user responsibilities and roles. Most users will work only within the local domain(s) that contain their area of responsibilities, and may not need to be aware of the Global Domain environment. For performance and user contention reasons, Global Domain usage should be limited to relatively infrequent processes that require data from multiple local domains.

The other primary feature of Global Domain is centralized configuration and administration. Most of the mechanisms required to build and administer a domain have been centralized and need only be run in the "master" domain, which either propagates data to the local domains or stores it centrally so that the local domains reference it in the master.

> **Note:** For a Global Domain environment to function properly, it is required that all local domains are structurally identical.

Measure Data

In a global domain environment, measure data can be physically stored in two different ways:

1. Across the local domains
2. In the master domain

Measure data stored in local domains is split across domains based on a pre-determined level of a given hierarchy. This level is defined during the configuration process and is referred to as the "partition" level.

The base intersection of a measure (that is, what dimensions a measure contains) determines whether data is stored in the local domains or in the master domain. If the base intersection of a measure is above the "partition" level or it does not contain the hierarchy on which the Global Domain environment is partitioned, the data will be stored in the master domain. This type of measure is referred to as a "Global Domain measure," or a "Higher Base Intersection measure."

Consider a global domain environment where the partition-level is based on the Department dimension in the Product hierarchy. In this scenario, data for measures that have a base intersection in the Product hierarchy at or below Department (other hierarchies are irrelevant for this discussion) is stored in the local domain based on which Department the underlying position in the Product hierarchy belongs to.

Measures that have a higher base intersection in the Product hierarchy than Department (for instance, Division) or measures that do not contain the Product hierarchy (such as a measure based at Store-Week, no Product Hierarchy) cannot be split across the local domains. These measures will reside in the master domain and will be accessed from there when these measures are required in workbooks.

All measures will be registered in the master domain and automatically registered in all local domains. RPAS automatically determines where the measure needs to be stored by comparing the base intersection of the measure against the designated partition-level of the Global Domain environment.

The physical location of the measure data will be invisible to the user after the measure has been registered. However, administrators must know where data for each measure is to be stored (master vs. local) as the data must be loaded in the proper location.

2
Domain Administration

Overview

The RPAS DomainDaemon is a process that is used to enable the communication channel between RPAS clients and RPAS domains.

The DomainDaemon runs on the server side and waits for requests from RPAS clients on a given port. Once it receives a request from a client, it starts a server process that the client connects to. From this point on, the client and server communicate directly. The system administrators may choose to have one single DomainDaemon process for all of the users, or they may choose to have separate processes per domain, per enterprise, and so on.

The DomainDaemon is installed in the [RPASDIR]/bin directory, where [RPASDIR] stands for the full path to the directory in which the RPAS server is installed. The system administrators can start, stop, and monitor the DomainDaemon processes using scripts that are provided in this directory.

Starting the DomainDaemon

In order to start the DomainDaemon, execute the script called DomainDaemon in the Acumate installation directory. The port number on which the DomainDaemon will be running must be passed in as an argument. The port number must be between 1025 and 65535. If 'auto' is specified instead of a number, the DomainDaemon is started on any available port.

> **Note:** In the following examples, [RPASDIR] stands for the full path to the directory in which the RPAS server is installed.

For example, issuing the following command from a UNIX shell will start a DomainDaemon on port 55278.

```
([RPASDIR]/bin)$ DomainDaemon –port 55278 -start
```

Monitoring the DomainDaemon

The –ping argument can be used to see whether a DomainDaemon is active. The port number must also be passed as an argument. If the DomainDaemon is active on the port, a message will be printed, and the script will return a value of 'true.' Otherwise, the script will return a value of 'false.'

Example

```
([RPASDIR]/bin)$ DomainDaemon –port 55277 –ping
DomainDaemon on port 55277 is alive.
```

Stopping the DomainDaemon

Use the –stop argument to stop the DomainDaemon running on a given port.

Example

```
([RPASDIR]/bin)$ DomainDaemon –port 55277 -stop
```

Other DomainDaemon Commands

add

Use the –add argument to add a domain to the list of domains managed by a DomainDaemon process. You must specify the port number and the complete path to the domain.

Example

```
DomainDaemon –port 55277 –add /mydomains/domain1
Added domain /mydomains/domain1 to daemon.
```

remove

Use the –remove argument to remove a domain from the list of domains managed by a DomainDaemon process. You must specify the port number and the complete path to the domain.

Example

```
DomainDaemon –port 55277 –remove /mydomains/domain1
Removed domain /mydomains/domain1 from daemon.
```

deactivate

Use the –deactivate argument to mark a domain as temporarily unavailable. You must specify the port number and the complete path to the domain.

Example

```
DomainDaemon –port 55277 –deactivate /mydomains/domain1
```

activate

Use the –activate argument to reactivate a domain that you previously deactivated. You must specify the port number and the complete path to the domain.

Example

```
DomainDaemon –port 55277 –activate /mydomains/domain1
```

showActiveServers

Use the –showActiveServers argument to list all the active server processes managed by the DomainDaemon. You must specify a port number.

For each active server, the DomainDaemon shows the process ID, domain, and user ID.

Example

```
DomainDaemon –port 55277 –showActiveServers
Registered Servers:
15920, /mydomains/domain1, adm
```

showDomains

Use the –showDomains argument to list the domains managed by the DomainDaemon. You must specify a port number. For each managed domain, the DomainDaemon indicates whether it is active or inactive.

Example

```
DomainDaemon –port 55277 –showDomains
Managed Domains:
/mydomains/domain1: Active
/mydomains/domain2: Inactive
```

showLockTable

Use the –showLockTable argument to display the contents of the database lock table.

Example

```
DomainDaemon –port 55277 –showLockTable
Lock Table:
```

releaseLocks

Use the –releaseLocks argument to release all database locks held by the specified process. You must specify a port number and a process ID.

Example

```
DomainDaemon –port 55277 –releaseLocks 15920
All locks released.
```

stopServer

Use the –stopServer argument to stop a specified server process. You must specify a port number and a process ID.

Example

```
DomainDaemon –port 55277 –stopServer 15920
Stop Server succeeded.
```

Centralized Administration

Note: If a solution is built in a Global Domain environment, most administrative activities can only be performed in the "master" domain. This applies to RPAS administrative workbook templates and wizards as well as RPAS utilities that are run on the back-end against the domain.

Administrative Workbook Templates and Wizards

The following list includes the standard RPAS workbook templates and/or wizards that have been centralized and can only be run in the master domain of a Global Domain environment. Please see the individual sections for additional information.

- **Alert Manager dialog window** – results of the alert finder run on the global domain are collated and displayed:
  - Applies for all alerts registered in the global domain
  - Results will be based on data from all the individual local domains
  - Results are consolidated (added together) to display a single result per measure
- **Alert Manager workbook template** – used to build alert workbooks from the Alert Manager dialog window. Data will be retrieved from the local domains.
- **Measure Analysis** – for analyzing measure data from local domains.
- **Security Administration** – ability to set security by template, measure, and positions. This workbook template can only be used in the master domain and has been disabled for use in local domains.
- **User Administration** – user information will be set up and maintained in the global domain, but will be replicated to the local domains. Updates will be effective immediately after the changes are committed. This workbook template can only be used in the master domain and has been disabled for use in local domains.
- **Translation Administration** – template used for modifying the labels of translatable data in RPAS. This workbook template can only be used in the master domain and has been disabled for use in local domains.
- **Hierarchy Maintenance** – for setting up and maintaining positions of user-defined dimensions (user-defined dimensions must be registered in the Global Domain using utility reguserdim).

RPAS Utilities

The following list includes the standard RPAS utilities that have been centralized and can only be run in the master domain of a Global Domain environment. See the individual sections for additional information.

- Alerts (alertmgr)
  - Alerts registered in the global domain will be propagated to local domains.

    Note: Alerts registered in local domains will not (and do not have to) be included in the global domain.

  - Utility for finding alerts (a.k.a. alert finder) will be updated to run against the local domains and collate results in the global domain for a centralized view of the alert results.
- Loading hierarchies (loadhier) – It is required that hierarchy information be centrally administered in the Global Domain and replicated to the local domains.
- Reshaping arrays (reshapeArrays) – Execution of this administrative utility in the global domain will perform this function on all local domains.
- Users (usermgr) – The back-end utility for managing users will propagate changes from the global domain to local domains. Ultimately, this will also be updated for the RPAS Administrative Template "User Administration."
- Domain properties (domainprop) – Manipulating properties, such as specifying password properties, locking user accounts, and so on.
- Miscellaneous registration utilities – Some of the registration utilities have been updated to operate in the global domain where all changes are propagated to the local domains:
  - Measures (regmeasure)
  - Templates (regtemplate)
  - Functions (regfunction)
  - Token measures (regTokenMeasure)
  - User-defined dimensions (reguserdim)

3

Security and User Administration

Functional Overview

This chapter describes the security model in RPAS, which includes workbook templates, workbooks, measures, and positions. The levels of security are defined as measure level, position level, and workbook level.

This chapter also describes user administration and security administration.

Note that if a solution is built in a Global Domain environment, it is only required and possible to perform the administrative activities included in this section in the “master” domain.

User Logon Security

A user account may be marked as **locked out** by the domain administrator.

This will prevent the user from logging in to the RPAS 11.0 client. The account remains locked out until the administrator re-enables the account.

Account lockouts may be set or cleared by the domain administrator by using the User Management utility.

A user account may be marked as **must change password**.

This is useful for brand-new accounts. The user will be allowed to logon with the current password, but will immediately be forced to select a new password.

Must change password may be set or cleared by the domain administrator by using the User Management utility.

Account Lockout may be enabled for a domain.

The domain administrator selects a number of failed logon attempts after which the User account will be marked as locked out. The account will remain locked out until the administrator re-enables it.

Account Lockout can be enabled through the domainprop utility using the –lockAccount flag.

Password expiration may be enabled for a domain.

The domain administrator selects a number of days after which passwords expire. When a user logs in, if the configured number of days have passed since this user entered a new password, the system requires a new password to be entered.

Password expiration can be enabled through the domainprop utility by using the –expirePassword flag.

Password history may be enabled for a domain.

The domain administrator selects a number of passwords to save. When a user attempts to change passwords, the system will not permit any password already stored in the password history to be used again.

Password history may be enabled through the domainprop utility by using the –passwordHistory flag.

Measure Level Security

Measures have access rights; which are read-write, read-only, or denied. Measures that are read-write or read-only may be selected in the extra measures and insert measure dialogs. RPAS ensures that read-only measures are not editable by the user and the presence of read-only measures does not affect the ability to commit a workbook.

Measure security can be specified when the measure is registered and can be changed later using the Security Administration workbook. The Measure Rights worksheet allows you to specify Read Only, Deny, or Read/Write access to a measure for each user.

A workbook template can override the default security of a measure, but can only narrow the security of the measure. For example, a measure could have default read-write access for a user and a template could specify that all users have read-only access to the measure when a workbook is built. However, if the default measure security was read-only, the template could not expand the security of that measure to read-write. Measures that are explicitly made read-only by a workbook template will not be expanded to read-write access by RPAS.

Position Level Security

Position Level Security allows access control for dimensions on a position-by-position basis. This capability is completely optional. If position level security is not explicitly defined and configured, all users in a domain have access to all positions in all hierarchies. Once position level security is defined, access to a position can be granted or denied for all users, for users in a group, or for an individual user.

Position level security can be defined at levels (dimensions) at or above base (such as class in the product hierarchy) in any hierarchy other than calendar. As positions are added at a level/dimension lower in the hierarchy than where the position level security is maintained, access to those positions is automatically granted if a user has access to the "parent" position. In other words, if security is maintained at the subclass level, users are automatically granted access to all the SKUs in a given subclass if they have access to that subclass, including those added after security has been established.

Exactly one dimension in each hierarchy can be defined as the security dimension for the hierarchy. If a security dimension is defined for the hierarchy, all dimensions in the hierarchy have position level security enabled, but position security is set at or above the dimension that is designated. For instance, if the "class" dimension is designated as the security dimension, an administrator can maintain access to positions in the class dimension or at any level above class.

To specify the security dimension for a hierarchy, use the RPAS Configuration Tools or the **inithier** utility. See the RPAS Configuration Tools User Guide or the section for

Defining Hierarchy Settings – inithier for more information.

After a security dimension is defined for a hierarchy, all users in the domain default to not having access to any positions in any dimension in the hierarchy. Worksheets in the Security Administration workbook are used to control position access for individual users, user groups, or all users (referred to as "world" or default access). There are three worksheets in this workbook for each hierarchy with a defined security dimension. One sheet controls access to positions for all users (default) (for instance, Prod Security Default); one controls access to positions by user group (for instance, Prod Security Group); and the third controls access to positions by individual user (for instance, Prod Security User).

Access must be granted at all levels for a user to have access to a position. This means that a position must have a value of true at the levels default/world, group, and user. The following table demonstrates how access is granted or denied based on all combinations of settings:

Security set by Position (Denied = False, Granted = True)			**Based on settings on left, user is Granted or Denied access**
User	**User Group**	**World**	**11.1**
Denied	**Denied**	**Denied**	**Denied**
Denied	**Denied**	**Granted**	**Denied**
Denied	**Granted**	**Denied**	**Denied**
Granted	**Denied**	**Denied**	**Denied**
Denied	**Granted**	**Granted**	**Denied**
Granted	**Denied**	**Granted**	**Denied**
Granted	**Granted**	**Denied**	**Denied**
Granted	**Granted**	**Granted**	**Granted**

Position level security is utilized when you select positions in the wizard process before building a workbook. Only positions to which a user has access are available for selection in the 2-tree, which are then included in the build of the workbook.

Workbook Security

Currently, workbook access is either granted or denied. If users have been granted access to a workbook; they can open, modify, and commit the workbook. No distinction is made between read-write-commit, read-write, and read-only access. Workbook access is automatically granted to the user that built it, and it may be shared with multiple groups or the world.

> **Note:** A user must have access to the workbook template in order to access the workbook, even if the workbook has world or group access rights.

Users with administrator status automatically have access to all workbook templates. By default, this means that administrators have access to all workbooks saved with world access. If a workbook is saved with group access, administrators can access the

workbook only if they are members of the default user group of the user who saved the workbook.

Another aspect of workbook security is the ability to set limits for the number of workbooks that a user can have saved at any given time. Limits can be set for a user per template, for a user group per template, or for a template for all users. The limits are evaluated in the above order, meaning that a limit defined at user-template overrides any values defined at group-template or template. If the above limits are not defined, the default value is one billion.

The limits are checked when a user begins the workbook build process. If the user's limit has been reached, an error message appears that informs the user that the workbook build process cannot complete because the user has reached their limit and what that limit is. The wizard process then terminates.

User Administration

Overview

User administration is the process by which administrators add and/or delete authorized system users, create and/or delete user groups, and edit user profiles. These tasks are performed through completion wizards on the User Administration tab. The following procedures are discussed in this area:

- Access the User Administration tab
- Add a user
- Add a user group
- Delete a user
- Delete a user group
- Edit a user's profile

Once users and user groups are set up, you can set access permissions to workbook templates and to measures within workbooks through Security Administration. Additionally, you can modify the label, default workbook template, and/or Admin status associated with individual users.

Procedures

Access the User Administration Tab

1. Select New from the File menu. The New dialog box is displayed.
2. Select the User Administration tab.

Add a User

1. From the File menu, select New.
2. Click the User Administration tab.
3. Select Add User and click **OK**.
4. In the ID field, type the ID string that the user will use for logging on.

   Note: Each user ID must begin with a letter, must contain no spaces (the underscore character is acceptable), and must not exceed eight characters total.

5. In the User Label field, type a label describing the user (for example, the user's full name). This identifying label will appear in various locations throughout the application. For example, labels appear on the File > Open dialog box to identify the owner of a given workbook and on the Forecast Approval worksheet to specify which user approved a given forecast.
6. In the Default Group field, select the user group to which the user will belong.
7. If a user will belong to more than one group, select the additional groups from the list in the Other Groups field.

8. In the Password field, type a password for the user.
9. In the Password Verification field, type the same password.
10. If the user should have Admin status, which allows that user to create system-wide default styles for workbook templates, check the Administrator box.

    Note: Admin status enables users to perform the Format menu option Save Format/Admin, which creates new system-wide default styles for workbook templates. If you are not sure whether a user should be granted this ability, note that a user's Admin status can later be modified on the Users worksheet of the User & Template Administration workbook.

    Note: While granting a user Admin status gives them access to all workbook templates, it does not automatically give them access to all workbooks.

11. If the user must change his or her password when logging on for the first time, check the Force Password Change box.
12. If you want to temporarily disable the user's account, check the Lock User Account box.
13. Click **Finish** to add the new user to the database.

You can now assign workbook template and measure access rights to the user. To do so, access the User & Template Administration workbook.

Add a User Group

User groups provide an intermediate level of security to workbooks created and saved by specific users. When you assign new users to the system, they must be assigned to existing user groups. User groups should consist of individuals with similar job functions or responsibilities. In the Oracle Retail Predictive Planning Suite, the user group corresponds to the user's planning role.

1. Select New from the File menu.
2. Click the User Administration tab.
3. Select Add User Group and click **OK**.
4. In the Group Name field, type a name for the group.
5. In the Group Label field, type a descriptive label for the group. This label is displayed when referring to the group throughout RPAS.
6. Click **Finish** to add the user group to the database.

Delete a User

If a user profile is no longer needed, it should be deleted from the system in order to maintain system security.

1. From the File menu, select New.
2. Click the User Administration tab.
3. Select Delete User and click **OK**.
4. Select the name of the user that you want to delete.
5. Click **Finish** to delete the user from the system.

Delete a User Group

If a user group no longer exists, you should delete the group from the system as soon as possible to maintain system security.

> **Caution:** Deleting a user group will delete every user in that group.

1. From the File menu, select New.
2. Click the User Administration tab.
3. Select Delete User Group and click **OK**.
4. Select the user group that you want to delete.
5. Click **Finish** to delete the user group from the system.

Edit a User

1. From the File menu, select New.
2. Click the User Administration tab.
3. Select Edit User and click **OK**.
4. Select the user you want to edit and click **Next**.
5. Make the necessary changes to the user's profile. You can change anything except the User Name. See "Add a User" for details.
6. Click **Finish** to save the changes you have made.

Security Administration Workbook

Overview

The Security Administration Workbook is only available to system administrators. After users and user groups are created, you may set up and maintain access permissions to workbook templates and measures within those workbook templates. That is, you can determine which templates individual users can access, as well as the measures that users can access while manipulating workbooks in the system. Additionally, you can specify and restrict the measures that are available to be added to a given workbook template. The ability to set access permissions in this way provides a high degree of measure security, as users can be restricted to viewing and editing only certain relevant measures.

The Security Administration workbook has one or more worksheets for each of the following:

- Workbook Template Rights Worksheet
- Workbook Template Measure Rights Worksheet
- Measure Rights Worksheet
- Dimension Modification Rights
- Position Level Security
- Workbook Template Limits

Security Template Administration also allows you to modify the label, Admin status, and/or default workbook template associated with each user. You also access this workbook template to modify the labels associated with user groups, workbook templates, and workbook template groups. Using this workbook, you can:

- Assign/modify access rights of each user to all workbook templates. User/template permissions are set in the Workbook Template Rights worksheet.
- Determine which optional measures are to be accessible through individual workbook templates. Template/measure permissions are set in the Workbook Template Measure Rights worksheet.
- Assign/restrict user access to individual measures. User/measure permissions are established in the Measure Rights worksheet.

Workbook Template Rights Worksheet

The Workbook Template Rights worksheet is for setting and maintaining access permissions of each user to specific workbook templates.

The worksheet contains a checkbox for each available workbook template and user combination. A checkmark in the cell indicates that the user has access rights to that specific template.

To grant a user access rights to a workbook template, put a checkmark in the checkbox in for that workbook template.

To deny a user access rights to that specific workbook template, leave the checkbox blank or clear the checkmark.

After changing a user's profile, the changes must be committed to the database in order for them to take effect.

Workbook Template Measure Rights Worksheet

The Workbook Template Measure Rights worksheet allows administrators to determine which registered measures will be available for optional inclusion in newly built workbooks.

When a measure is initially registered as a public measure, all templates default to having access to that measure. This means that it is possible for this measure to be added to a workbook template even if it is not one of the standard measures displayed when a workbook of that type is built. Some new workbook wizards include a dialog that prompts users to select any additional measures to be included in the workbook build. By default, all newly registered measures are included on this list of available additional measures. The other method of inserting new measures into a workbook is via the Insert Measure command. See "

Insert Measures into an Open Worksheet" on page 4 for more information.

The Workbook Template Measure Rights worksheet is used to modify template/measure permissions, thus allowing only certain templates to optionally include specified measures in new workbook builds.

This worksheet contains a checkbox for each available workbook template and registered measure combination. A checkmark in the cell indicates that the measure will appear on the additional measures list in the new workbook wizard, and is thus available for inclusion in the initial workbook build.

To make a given measure available in a workbook template, put a checkmark in the checkbox that represents that template/measure intersection.

To prevent a measure from appearing in a specific workbook template, clear the checkbox located at that template/measure intersection.

After changes are made to a template's profile, they must be committed in order for them to take effect.

Note: Measures that are registered as private measures will not appear in this worksheet. If there are no public measures available to be displayed in this worksheet, the worksheet will not be built.

Measure Rights Worksheet

The Measure Rights worksheet allows you to restrict user access to individual measures on a user-by-measure basis. User/measure permissions are initially determined by the system by integrating the current user/template and template/measure settings and applying the following rule:

"A user cannot have access to any measure that is not available in at least one template to which the user has access."

Permissions can be made even more restrictive on a user by measure basis by using the Measure Rights worksheet to deny users access to measures that they would normally be permitted to edit.

The worksheet contains a checkbox for each available user and registered measure combination. A checkmark in the cell indicates that the user has access rights to and is permitted to view and edit the specified measure. More specifically, that given user has access to at least one template that is permitted to include the specified measure.

You can further restrict a user's access to specific measures by removing checkmarks that are displayed at those associated intersections. Removing the checkmark (by clicking on it) prevents the user from accessing a given measure, even though that measure is normally permitted to be displayed in a particular workbook template to which the user does have access.

After changes are made to a user's profile, they must be committed in order for them to take effect.

> **Note:** The Measure Rights worksheet contains only public measures; that is, measures that can be optionally included in a worksheet, depending on choices made in a new workbook wizard. Measures that are registered as private measures will not appear in this worksheet. If there are no public measures available to be displayed in this worksheet, the worksheet will not be built.

Dimension Modification Rights Worksheet

The Dimension Modification Rights worksheet allows you to determine which dimensions, if any, a user can modify. The worksheet contains a checkbox for each available user and dimension combination. A checkmark in the cell indicates that the user is permitted to modify the specified dimension.

After changes are made to a user's dimension modification rights, they must be committed in order for them to take effect.

Position Level Security Worksheets

The position-level security worksheets are used to grant or deny access to positions for individual users, user groups, or all users. Position-level security is set for a specific dimension of a hierarchy (other than calendar). This is completed using the **inithier** utility or the Configuration Tools.

For each hierarchy/dimension for which position level security is enabled (normally just a single hierarchy/dimension) there are three worksheets: one each for user, user group, and world/all users.

After changes are made to position level security they must be committed in order for them to take effect.

The Workbook Template Limits worksheets are used to limit the number of workbooks that a user can have saved. Limits can be set for a user per template, for a user group per template, or for a template for all users. The limits are evaluated in the above order, meaning that a limit defined at user-template overrides any values defined at group-template or template. If the above limits are not defined, the default value is one billion.

The limits are checked when a user begins the workbook build process. If the user's limit has been reached, an error message appears that informs the user that the workbook build process cannot complete because the user has reached their limit and what that limit is. The wizard process then terminates.

Procedures

> **Note:** These tasks are performed through the Security Administration Workbook. This workbook is only available to system administrators.

Access Security Administration

1. From the main menu, select File > New. The New dialog box is displayed.
2. Select the Administration tab to display a list of workbook templates for Administration.
3. Highlight Security Administration and click **OK**.

Set or Modify Users' Access to Workbook Templates

1. From the File menu, select New.
2. Click the Administration tab.
3. Select Security Administration and click **OK**.
4. On the Workbook Template Rights worksheet, select each template for which a user needs access rights. For templates to which the employee should not have access rights, make sure there is no check mark.
5. Changes must be committed to the master database for them to take effect. To commit your changes, select Commit Now from the File menu.
6. If you desire, save your workbook by selecting Save from the File menu.
7. To close the workbook, select Close from the File menu.

Set Measure Availability for Workbook Templates

1. From the File menu, select New.
2. Click the Administration tab.
3. Select Security Administration and click **OK**.
4. On the Workbook Template Measure Rights worksheet, select each registered measure that should be available for inclusion in the associated workbook template. For measures that should not be included in the associated template, make sure there is no check mark.
5. Changes must be committed to the master database for them to take effect. To commit your changes, select Commit Now from the File menu.
6. If you desire, save your workbook by selecting Save from the File menu.
7. To close the workbook, select Close from the File menu.

Assign or Restrict User Access to Measures

1. From the File menu, select New.
2. Click the Administration tab.
3. Select Security Administration and click **OK**.
4. On the Measure Rights worksheet, for each measure that a user should have access to, select either Read Only or Read/Write from the drop-down list. For measures to which the user should not have access, make sure Denied is selected.
5. Any changes made must be committed to the master database in order for them to take effect. To commit your changes, select Commit Now from the File menu.
6. If you desire, save your workbook by selecting Save from the File menu.
7. To close the workbook, select Close from the File menu.

Change a User's Ability to Modify Dimensions

1. From the File menu, select New.
2. Click the Administration tab.
3. Select Security Administration and click **OK**.
4. On the Dimension Modification Rights worksheet, select each dimension for which the user needs modification rights. For dimensions that the user should not be able to modify, make sure there is no check mark.
5. Any changes made must be committed to the master database in order for them to take effect. To commit your changes, select Commit Now from the File menu.
6. If you desire, save your workbook by selecting Save from the File menu.
7. To close the workbook, select Close from the File menu.

Set or Modify Access to Positions (if position level security has been enabled)

1. From the File menu, select New.
2. Select the Administration tab.
3. Select Security Administration and click **OK**.
4. Select the worksheet for which security needs to be set or modified: User, User Group, or World.
5. By default, the dimension (level) at which position level security has been enabled will be displayed. To manage security at a level above the designated level (only levels above are possible), right-click and "Select Rollup" to view the available dimensions.
6. To grant access to a position, click the checkbox of the cell. Note that a user must have access at the User, User Group, and World levels to have access to a position.
7. Changes must be committed to the domain before exiting to take effect.

Limit the Number of Workbooks that a User can have Saved

1. From the File menu, select New.
2. Select the Administration tab.
3. Select Security Administration and click **OK**.
4. Select the worksheet for which the limit will be set: User / Template, Group / Template, or Template.
5. Set the values as necessary.
6. Commit the data to the domain before exiting

4

Hierarchy Maintenance

Overview

Hierarchy Maintenance Workbook

Oracle Retail Predictive Solutions provide the ability to set up and maintain user-named and user-defined dimensions within hierarchies. Hierarchy Maintenance is the means by which custom-created dimensions within a hierarchy can be established and maintained through the application interface to meet individual business needs.

When Oracle Retail Predictive Solutions are installed, implementation scripts define the dimensions and hierarchical structures specific to your organization. For example, the system can be built to recognize that SKUs roll up into styles, styles roll up into product classes, and so on within the product hierarchy. Occasionally, you might want to group products according to some ad hoc personal design to suit a particular business need. You can group arbitrary items in a hierarchy to use in functions; such as forecasting, replenishment, and measure analysis. These user-defined groupings act as normal dimensional levels; that is, they allow you to roll data up from lower levels of aggregation along the hierarchical paths that you define.

Example

Suppose that your experience has shown that the accuracy of forecasts for your top 50 products (A products) reflects the relative accuracy of all forecasts. Therefore, you would like to group elements within a user-defined dimension as the top 50 products by designating them ‘A Products.’ When you select products in a wizard or look at data in a worksheet, you can change the rollup to your user-defined dimension to see your top 50 products grouped together.

Note that your collection of 50 products may comprise elements from a wide range of product classes or departments, and that your grouping scheme may have little to do with the normal dimensional relationships of these items in the product hierarchy. Note also that the group of items you designate as ‘A Products’ may change over time as consumer preferences change. From this example, you see that user-defined dimensions can be used to create any ad hoc groupings to provide additional support in analyzing, selecting, or summarizing data in Demand Forecasting. Moreover, the Hierarchy Maintenance interface allows you to change the nature of the groupings as required.

> **Note:** The number and names of user-definable dimensions are set by your company when Demand Forecasting is initially installed. The positions within each dimension and their associated labels can be altered and maintained through the hierarchy maintenance process.

Keep in mind that any hierarchy in the Oracle Retail Predictive Solutions can have user-defined dimensions within it (if set up by your company at the time of installation). The examples in this section refer to the Product hierarchy, but other hierarchies could be maintained in the same way.

Hierarchy Maintenance Example

Suppose you want to designate SKUs in your product hierarchy as either A, B, or C products so that you can group these items together when you view information such as forecasting, replenishment, or measure analysis reports.

To do this, you need to maintain a user-defined dimension that will allow you to map the SKUs to the various positions of your classification scheme (A, B, or C). The user-defined dimension used in the following example is named Product Status. To maintain this user-defined dimension, you use the Hierarchy Maintenance Wizard.

Hierarchy Maintenance Wizard

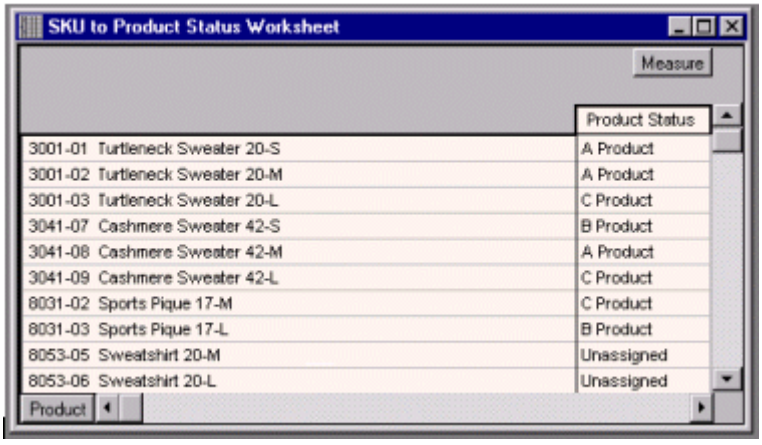
The first step in maintaining hierarchies is to access the Hierarchy Maintenance Wizard. In this wizard, you select the SKUs that will be mapped to the various positions of your user-defined dimension. Responses to prompts in the wizard are used to format a new Hierarchy Maintenance Workbook.

Hierarchy Maintenance Worksheet

The Hierarchy Maintenance worksheet displays the position assignment fields for your selected custom dimension. You can edit the cells associated with your custom dimension as required.

Returning to the example dimension Product Status, you want to classify each selected SKU in your workbook as an A Product, a B Product, or a C Product. This example provides only three positions, or values, in the Product Status dimension. However, you can enter any character string in an individual SKU's Product Status cell, and this new string will be treated as a separate user-defined grouping. If this is the first time a particular SKU has been mapped to the Product Status dimension, the label assigned to that SKU will not yet be defined. The Product Status field is automatically filled with 'Unassigned.'

Assign labels to each product with regard to the Product Status dimension. In the example below, products that were previously 'Unassigned' are now designated as A, B, or C Products.

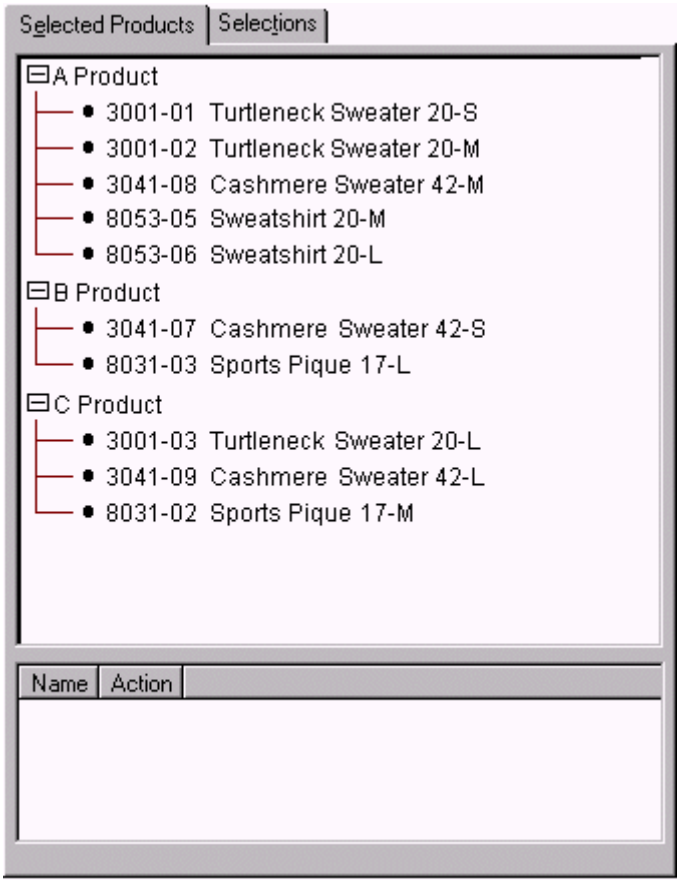


Note: The Oracle Retail Predictive Solutions system is case-sensitive when a new position name (label) is entered in the Hierarchy Maintenance Workbook. After the workbook has been committed, the typing of the group name is not case-sensitive. For example, "B Product" can later be entered as "b product" after the "B Product" group label has been committed.

After making the A, B, or C Product designations for the selected SKUs, you must commit the workbook for any changes to take effect.

For this example, labels have now been assigned to the various positions within the Product Status dimension, and selected products in the product hierarchy have been classified with regard to the custom dimension. Demand Forecasting treats Product Status, a user-defined dimension, as a normal dimensional level within the product hierarchy.

The figure below displays the results when, in a wizard, you access a quick menu and change the rollup to the Product Status dimension. The products shown here are classified according to the position values (A Product, B Product, or C Product) that you assigned while maintaining the Product Status dimension.



Procedures

Access Hierarchy Maintenance

Note that if a solution is built in a Global Domain environment, it is only required and possible to perform all of the administrative activities in this section in the “master” domain.

1. Select Open from the File menu to bypass the Hierarchy Maintenance wizard and open an existing Hierarchy Maintenance workbook.

   OR

2. Select New from the File menu.
3. Select the Administration tab to display the list of Administration templates.
4. Select Hierarchy Maintenance and click **OK**.
5. Select the hierarchy for which you want to specify a user-defined dimension (for example, Product or Location). Only the hierarchies that have been set up to contain user-defined dimensions are represented here.
6. Click **Next**.
7. Select the user-defined dimension to be updated. The number and names of available custom dimensions are set at installation.
8. Click **Next**.
9. On the Available side of the selection wizard, choose the items to be mapped to positions within your custom dimension. Click the right arrow button to move them to the selected side. When you have chosen all items to appear in your workbook, click **Finish**.

Maintain a User-Defined Dimension within a Hierarchy

Use this procedure to assign product or location items to custom-defined positions within a specialized dimension. Custom-created dimensions are distinct from those in the standard hierarchical roll-ups established at your system implementation. However, you can use these dimensions like normal Demand Forecasting levels, aggregating data along these new hierarchical paths.

1. Select New from the File menu.
2. Select the Administration tab to display the list of Administration templates.
3. Select Hierarchy Maintenance and click **OK**.
4. Select the hierarchy for which you want to specify a user-defined dimension (for example, Product or Location). Only the hierarchies that have been set up to contain user-defined dimensions are represented here.
5. Click **Next**.
6. Select the user-defined dimension to be updated. The number and names of available custom dimensions are set at installation.
7. Click **Next**.
8. On the Available side of the selection wizard, choose the items to be mapped to positions within your custom dimension. Click the right arrow button to move them to the selected side. When you have chosen all items to appear in your workbook, click **Finish**.

9. The Hierarchy Maintenance workbook is displayed. In the position assignment field for your custom dimension, assign a value to each product or location position in your workbook. You can enter any text string in a cell. Each unique string will be treated as a separate user-defined position within the custom dimension.
10. Select Commit Now from the File menu to commit your changes to the master database. If desired, you may also save your workbook by selecting Save from the File menu.
11. To close the workbook, select Close from the File menu.

5
Measure Analysis

Overview

Measure Analysis Workbook

The Measure Analysis Workbook template allows you to view data associated with any registered measure in the Oracle Retail Predictive Solutions applications, such as actual sales data for specified product/location/calendar combinations. You may also use the Measure Analysis workbook to edit values for read-write measures.

Although a common use of the Measure Analysis Workbook is to view actual sales data, the workbook is not restricted to presenting sales data alone. You can view any data loaded into the Oracle Retail Predictive Solutions master database, such as selling prices, shipments, orders, and so on. The Measure Analysis Wizard provides you with a list of all registered measures from which to select. Just choose the measures to be displayed in the new workbook.

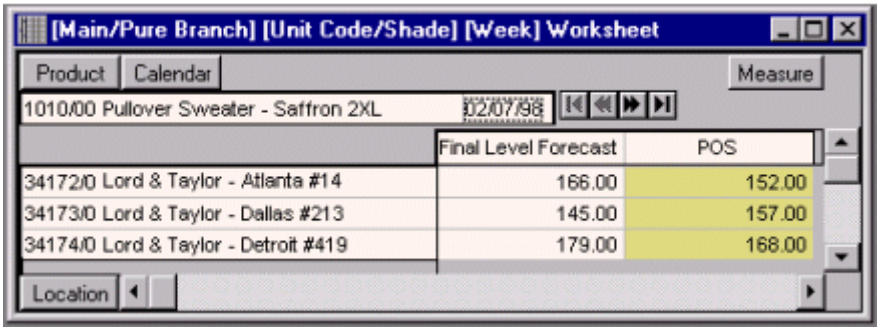
Measure Analysis Wizard

The Measure Analysis Wizard guides you through the process of creating a new Measure Analysis workbook, in which you can view data associated with any registered measure(s) in Demand Forecasting.

Measure Analysis Worksheet

The Measure Analysis workbook displays data associated with selected registered measures, and contains one type of worksheet, the Measure Analysis worksheet. A separate worksheet will exist for each distinct product/location intersection associated with the measures selected in the wizard.

The Measure Analysis worksheet allows you to view the chosen measure data for the selected products, locations, and time periods. Each Measure Analysis worksheet is displayed at a different dimensional intersection, depending on the measure selections made in the wizard. This dimensional intersection is shown in the worksheet title bar.



The example above shows a Measure Analysis worksheet that displays Final Level Forecast data and POS data for three locations. The location/product/calendar dimensional intersection of this worksheet, as shown in the title bar, is [Main/Pure Branch][Unit Code/Shade][Week]. The Final Level Forecast measure, because it is registered as a read/write measure, can be edited in this worksheet. POS values cannot be changed as this measure is registered as read only.

Procedures

Access Measure Analysis

1. Select Open from the File menu to bypass the Measure Analysis wizard and open an existing Measure Analysis workbook.

   OR

2. Select New from the File menu.
3. On the Analysis tab, select Measure Analysis and click **OK**.

Review and Edit Sales or other Registered Measure Data

1. To open an existing Measure Analysis workbook, select Open from the File menu, double-click on the workbook to be opened, and go to step 9.

   OR

1. To open a new workbook, select New from the File menu.
2. On the Analysis tab, select Measure Analysis and click **OK**.
3. The Measure Analysis Wizard opens and prompts you to select the registered measures to be displayed in the new workbook. Use Ctrl-Click and/or Shift-Click to select multiple measures.
4. Click **Next**.
5. Select the dimensional level at which you want to view measure data.
6. Click **Next**.

> **Note:** This screen only appears if there is more than one level possible for one of your selected measures. A separate wizard screen will be displayed for every measure chosen in step 3 if multiple aggregation levels are available.
>
> **Tip:** Selecting a pre-aggregated level (a level other than base level) allows you to view aggregated sales data faster than if you select the base level and later aggregate data from within the workbook itself. Building the workbook with pre-aggregated data can save time and lessen the need for data transfers later.

7. Select all calendar periods for which you want to view data.
8. Click **Next**.
9. Select the locations whose measure data you want to view.
10. Click **Next**.
11. Select the products whose measure data you want to view.
12. Click **Next**.

13. Click **Finish** to open the Measure Analysis workbook.
14. On the Measure Analysis Worksheet(s), view the stored data associated with the measures, products, locations, and calendar periods you selected. Make any changes as required.
15. Commit your changes to the master database by selecting Commit Now from the File menu. If desired, save your workbook by selecting a Save option from the File menu.
16. To close the workbook, select Close from the File menu.

6

Workbook Auto Build Maintenance

Overview

The Workbook Auto Build feature allows users to set up workbook builds to take place on a regular basis during nightly batch runs. Workbooks to be built in this way are added to the auto build queue. Because the workbook build process is automated, users are spared the processing time required to regularly enter the same wizard selections each time a new workbook is built. Because the build process occurs overnight, users are spared the wait time associated with constructing new workbooks.

The Workbook Auto Build feature works through the Workbook Auto Build Maintenance Wizard.

Workbook Auto Build Maintenance Wizard

The Workbook Auto Build Maintenance wizard steps you through the processes of adding and/or deleting workbooks from the auto build queue.

Procedures

Accessing the Workbook Auto Build Maintenance Workbook

1. Select New from the File menu.
2. Select the Administration tab.
3. Highlight Auto Workbook Maintenance and click **OK**.

Add a Workbook to the Auto Build Queue

Workbooks in this queue are designated to be built automatically on a specified regular basis as part of the nightly batch run.

1. Select New from the File menu.
2. Select the Administration tab.
3. Highlight Auto Workbook Maintenance and click **OK**.
4. From the task list, select Add Workbook and click **Next**.
5. Select a workbook template type and click **Next**.
6. Select an owner for the workbook and click **Next**.
7. Fill in the workbook Build Label, the Build Frequency (in days) with which the workbook should be built, and the Next Build Date.
8. Specify the Saved Access for the workbook: select User, Group, or World.
9. Select the group that owns the workbook. You can choose from the list of groups to which you belong.
10. Click **Next** to initialize the wizard for the workbook template selected in step 5 above. The choices you make are saved under the name you specified for the Build Label.

Delete a Workbook from the Auto Build Queue

1. Select New from the File menu.
2. Select the Administration tab.
3. Highlight Auto Workbook Maintenance and click **OK**.
4. From the task list, select Delete Workbooks and click **Next**.
5. Select the workbook or workbooks to delete from the auto build queue.
6. Click **Finish** to delete the workbooks from the Auto Workbook Build queue.

7
Translation Administration

Overview

All of the Oracle Retail Predictive Solutions can be presented in multiple languages. Every product, location, and calendar position can be displayed in multiple languages as can messages presented through the client. Translation administration is the process by which administrators maintain the translation tables for system text, including workbook template labels, template group labels, user group labels, wizard instructions, pick list options, and error messages.

The Translation Administration workbook contains worksheets for translating text used in measure labels, workbook template names, template group names, user group labels, and general areas (wizard instructions, pick list options, error messages, and so on).

The worksheets contained in the Translation Administration workbook include:

- General Translations worksheet
- Measure Translations worksheet
- Template Translations worksheet
- Template Group Translations worksheet
- User Group Translations worksheet

General Translations Worksheet

The General Translations worksheet allows you to view and edit the translations of text strings found throughout the Oracle Retail Predictive Solutions. Translations into each of the system's alternative languages are supported for all wizard instructions, pick list options, system warnings, error messages, menu options, and so on.

Measure Translations Worksheet

The Measure Translations worksheet allows you to view and edit the translations of measure labels. Translations are supported for each of the system's allowable alternative languages.

Template Translations Worksheet

The Template Translation worksheet allows you to view and edit the translations of workbook template names. Translations are supported for each of the system's allowable alternative languages.

Template Group Translations Worksheet

The Template Group Translations worksheet allows you to view and edit the translations of template group names. Translations are supported for each of the system's allowable alternative languages. Translations in this worksheet affect the labels on the tabs that appear in the File > New dialog; for example (in English), Administration, Analysis, and Predict.

User Group Translations Worksheet

The User Group Translations Worksheet allows you to view and edit the translations of user group labels. Translations are supported for each of the system's allowable alternative languages. The list of user groups includes the Administration, Default, and Internal user groups, plus any other user group names set up by the system administrator. For products in the Oracle Retail Predictive Planning Suite, the list of user groups also includes the various planning roles.

Procedures

Accessing the Translation Administration Workbook

1. Select New from the File menu.
2. Select the Administration tab.
3. Highlight Translation Administration and click **OK**.

Edit the Foreign Language Translations of Labels and System Text

1. Select New from the File menu.
2. Select the Administration tab.
3. Highlight Translation Administration and click **OK**.
4. Maximize the appropriate worksheet:
   - To translate measure labels, maximize the R_TRANS_MEAS worksheet.
   - To translate workbook template names, maximize the R_TRANS_WBT worksheet.
   - To translate template group names, maximize the R_TRANS_WBTG Translations worksheet.
   - To translate user group labels, maximize the R_TRANS_GRP worksheet.
   - For all other translations, such as wizard instructions, list options, error messages, menu choices, and so on, maximize the R_TRANS_MSG worksheet.
5. At the intersection of the language and the text value to be translated, enter the correct translated text. Repeat this step as required for all text strings to be edited.
6. On the File menu, select Commit Now.
7. On the File menu, select Close. Select a close option from the Close dialog.

Process for using the General Translations Worksheet

Enter the correct translated text at the intersection of the language and the string value to be translated.

Process for using the Measure Translations Worksheet

Enter the correct translated text at the intersection of the language and the measure label to be translated.

Process for using the Template Translations Worksheet

Enter the correct translated text at the intersection of the language and the workbook template name to be translated.

Process for using the Template Group Translations Worksheet

Enter the correct translated text at the intersection of the language and the template group label to be translated.

Process for using the User Group Translation Worksheet

Enter the correct translated text at the intersection of the language and the user group label to be translated.

8

Commit as Soon as Possible

Overview

Commit As Soon As Possible (Commit ASAP) allows users to schedule the commit process of workbook data so that it executes as soon as all the system resources are available. Commit ASAP is an option in the File menu of the RPAS client. Procedures for using Commit ASAP are provided in the "RPAS 11.1 Users Guide."

Commit ASAP takes a copy of the data to be committed. Unlike Commit Later (which adds a workbook commit process to a queue that is run in batch), the data that is eventually committed is the data that was present at the time the commit instruction was issued. With Commit Later, if the user makes further changes to the workbook and saves that workbook before the batch commit process is run, those changes will also get committed.

Using Commit ASAP

After attempting to commit a workbook using Commit ASAP (File\Commit ASAP), the user will see a message in the client that the workbook has been scheduled for a commit. The user can continue with their work. The system will then try to commit the workbook as soon as it can, taking into account any other scheduled commits. If the commit cannot be done prior to the domain's Commit ASAP deadline, it will be canceled and listed as failed.

There are four states for commit processes added to the Commit ASAP queue.

- Pending – the commit process is queued up to take place at some point in the future
- Committing – the workbook is currently being committed.
- Success – the commit succeeded
- Failed – the commit failed

The status of each commit ASAP process can be viewed using a dialog window called "**Commit Status**" (from the File menu). This dialog window displays all of the Commit ASAP processes with their respective status for all processes that have not been purged (see below). This dialog can be used to sort the tasks based on any of the columns.

Users can filter the entries in a variety of ways. If the checkbox **All Users** is not checked, the user will see only their entries. If it is checked, they will see the entries for all users. The checkboxes in the **Status To Display** group allow you to filter the output so that you see only the processes with the specified statuses. The window can be updated using the **Refresh** button. Also note that the dialog remembers the settings based on the last use.

Important Notes

- If a user attempts to commit a workbook ASAP that already has a process in the queue, the original processes will be removed from the queue. That means that there can only ever be one pending commit ASAP in the queue for a given workbook/user/template name combination.
- Workbooks must have been saved at least once before attempting a Commit ASAP. A workbook has not been saved if the label says "untitled."

Managing the Workbook Queue – showWorkbookQueues

The RPAS utility **showWorkbookQueues** is used for viewing the status of Commit ASAP processes and for purging entries in the Commit ASAP status window. The usage of this utility follows below.

The purge option requires a date before entries will be removed. It also requires specification for which entries to remove: succeeded, failed, or both.

Usage

showWorkbookQueues -version

showWorkbookQueues -d *domainPath* –show [all|pending|waiting|working|success|failed]*

showWorkbookQueues -d *domainPath* -purge *date* [success | failed]*

Argument	Description
–version	Prints the RPAS version, revision, and build information of the utility.
-d *domainPath*	Specifies the path to the domain.
-show	Lists the contents of the queue in the order in which the parameter is specified. Possible values: all, pending, waiting, working, success, and failed.
All	Used with the –show parameter, this lists all of the workbooks in all statuses.
Pending	Used with the –show parameter, this lists all workbooks that are waiting to be committed.
Waiting	For Oracle Retail development use only.
Success	Used with the –show parameter, this lists all workbooks that have been successfully committed.
Failed	Used with the –show parameter, this lists all workbooks that did not successfully commit.
-purge *date*	Purges entries in the Commit ASAP status window. Entries before the date provided will be removed. The date should be a string of the following DateTime format: YYYYMMDDHHmm For example “200406071529” equals June 7, 2004 3:29 PM. Administrator must select to purge commit processes that either succeeded or failed.

Commit ASAP Settings – configCommitAsap

There are two settings for Commit ASAP that are managed by an administrator. Both are set using the utility **configCommitAsap**.

- Maximum number of simultaneous commit processes (property MaxProcesses, default value is 4).
- Deadline for which all pending processes must be completed, after which they will be cancelled and marked as failed.
  - This deadline will likely be used by administrators before beginning nightly batch processes (property **deadline**, default value is 00:01 [meaning 12:01 AM], in 24-hour time).
  - A commit process that starts before the deadline is reached will be processed. Commit requests that were in the queue before the deadline that did not get processed will be cancelled and marked as failed. Commit requests added to the queue after the deadline will use the deadline the following day.

Usage

configCommitAsap -d *pathToDomain* [-maxProcs *numProcs*]
[-deadline *time*]
[-display]

Argument	Description
–version	Prints the RPAS version, revision, and build information of the utility.
-maxProcs *numProcs*	Sets the maximum number of concurrent commit processes where numProcs is an integer greater than 0. Workbooks can be committed in parallel if they do not require access to the same measure databases. If they do share databases they will be committed sequentially.
-deadline *time*	The time of the day when all outstanding commit ASAP operations will timeout. If a commit ASAP operation is submitted after this time, it will not timeout until the deadline time on the next day. This string must have the following format: HH:MM For example “13:30” refers to 1:30 PM
-display	Displays the current commit ASAP settings.

Argument	Description
-loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.
-noheader	To disable timestamp header use.

Logging and Technical Information

A log file is available in the Commit ASAP directory that should be checked if a user reports an error with a Commit ASAP submission. The file is named **rpasServer.log** and is in the following directory: <Path to domain>/commitAsapQueue.

Another log file is generated for each Commit ASAP process and stored in a user's directory (users/<userid>/asapLogs). The format of the log file name is:

orig_<original workbook name>asap_<temporary workbook name>.log

RPAS creates a temporary workbook in this process to capture the snapshot of the data that needs to be committed. Temporary workbooks are never viewed by a user. An administrator can use this log if something does not properly commit. Note that these "snapshot" workbooks cannot be viewed or used in the RPAS client.

An example of this log file is orig_t1_asap_t5 where "t1" is the name of the original workbook and "t5" is the name of the snapshot workbook.

The following directories are used to store the copies of the workbook as they are processed through the system:

- pending directory – Contains one file per submitted commit ASAP that has not yet been processed. These files are, in general, binary and cannot be easily read.
- working directory – Contains one file per submitted commit ASAP that is currently in the commit process.
- success directory – Contains one file per submitted commit ASAP that has successfully completed its commit process.
- failed directory – Contains one file per submitted commit ASAP that either had a failure during its commit process or could not be committed prior to the deadline.
- unknown directory – If the Commit ASAP process detects a corrupted queue file, a message gets logged and the file gets moved into the unknown directory.

9

Batch Processes and RPAS Utilities

Overview

Included with an RPAS installation is a collection of stand-alone executables and scripts that are used for a variety of operations. RPAS utilities are run directly against a domain. If in a Global Domain environment, most utilities can only be run on the master domain. RPAS utilities can be categorized into the following groupings:

- Hierarchy management – the loading and refreshing of hierarchies, and the process of updating the data structures in the domain to reflect hierarchy changes
- Measure data – utilities for loading, exporting, and moving data within and between domains
- Miscellaneous – a variety of utilities for performing certain procedures in batch and for setting a number of parameters on an environment/domain
- Information RPAS utilities – a variety of utilities that retrieve information about a domain, data, the RPAS server code, or an object used by the server

Using Shell Scripts to Run Batch Processes

Batch processes should be written using scripts that call the RPAS 11 binaries found in the $RPAS_HOME/bin/ directory. Any log files generated by scripts will be in the [DOM]/scripts/err/ directory. Examples of tools include Korn shell, Python, and Perl.

A Sample Shell Script

The following is a sample shell script that loads the product and location hierarchies into a domain. It is assumed that this script is invoked from the [DOM]/scripts/ directory.

```
1  #!/bin/ksh
2  loadHier –d .. –load prod > ./err/loadhier.prod.log
3  loadHier –d .. –load loc >> ./err/loadhier.loc.log
```

Line 1 defines the shell that will execute the script. In this example, it is defined to be the Korn shell. Therefore, this script will always be executed from the Korn shell even if the user's shell is different.

Lines 2 and 3 call the loadHier utility to load the latest product and location hierarchy information. Depending on the batch process to be performed by the shell script, lines 2 and 3 can be replaced by one or more lines to call one or more RPAS 11 utilities.

Centralized Utilities

In a Global Domain environment, most of the RPAS utilities can only be run on the master domain. These changes are then propagated to the local domains or stored centrally for access by each local domain.

The following utilities are centralized:

- Alerts (**alertmgr**)
  - Alerts registered in the global domain will be propagated to local domains.

    Note: Alerts registered in local domains will not (and do not have to) be included in the global domain.

  - Utility for finding alerts (a.k.a. alert finder) will be updated to run against the local domains and collate results in the global domain for a centralized view of the alert results.
- Loading hierarchies (**loadhier**) – It is required that hierarchy information be centrally administered in the Global Domain and replicated to the local domains.
- Reshaping arrays (**reshapeArrays**) – Execution of this administrative utility in the global domain will perform this function on all local domains.
- Users (**usermgr**) – The back-end utility for managing users will propagate changes from the global domain to local domains. Ultimately, this will be updated for the RPAS Administrative Template "User Administration."
- Domain properties (**domainprop**) – Manipulating properties, such as specifying password properties, locking user accounts, and so on.

Common Information and Parameters for RPAS Utilities

A number of standard parameters are available for most RPAS utilities. Check the usage of a specific utility to verify whether or not it is available.

Argument	Description
–version	Use this argument to get the version information of the utility (for instance, RPAS 11.1.0). It does not require –d domainPath.
–d *pathtodomain*	Specifies the path to the domain against which the utility will run.
–loglevel	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.
–n	Certain utilities contain this parameter to perform a dry run. Using this option will show the administrator what would change but makes no actual changes to the system or data. See the usage of a specific utility to see whether this option is applicable.

Argument	Description
–noheader	Used to disable the use of a timestamp in the header of the log file.

Logging Batch Processes

RPAS batch processes may run for a long time. Therefore, it is sometimes difficult to know the step currently being performed. Fortunately, the RPAS server code provides a large amount of detailed output messages from its processes to help the system administrator troubleshoot, benchmark, and estimate job completion times. The batch processing shell scripts direct those messages to output files (log and error files) located in [DOM]/scripts/err. These output files typically use the following naming convention:

batchProcess.date&timestamp.fileType

where:

BatchProcess = The name of the shell script or the corresponding batch process.

data&timestamp = The date and time when the script is executed in the format yyyymmddhhmmss.

fileType = The type of the file that can be “log” or “err” to indicate whether it is a log file or an error file.

Configuration Tools Log Files

For the RPAS Configuration Tools, information is logged in the files **stderr.txt** and **stdout.txt**, which are located in the **bin** sub-directory of the Tools directory. If you experience a problem with the configuration tools, you should send these two files to Oracle Retail Customer Care along with a description of the problem.

Error Files

Error files are usually generated during data loading. These files include a list of bad records followed by the total number of records read and related information. Here is part of the output from a typical data loading batch job.

```
Loading array TEMP in nonoverlay mode. Zero values are loaded.
Skipping cell with invalid position:
    INFO    1
    DAY 1996D364
    SKU SKU_00726828
    STR STR_0107
Skipping cell with invalid position:
    INFO    1
    DAY 1996D364
    SKU SKU_00726828
    STR STR_0201
Skipping cell with invalid position:
    INFO    1
    DAY 1996D364
    SKU SKU_00726828
    STR STR_4008
Skipping cell with invalid position:
    INFO    1
    DAY 1996D364
    SKU SKU_00726828
    STR STR_4009
```

```
Skipping cell with invalid position:
    INFO   1
    DAY 1997D6
    SKU SKU_00726828
    STR STR_0107
Skipping cell with invalid position:
    INFO   1
    DAY 1997D6
    SKU SKU_00726828
    STR STR_0201
Array -- TEMP. Load time:   0:02
LoadComplete -- Records read: 4240, Total cell updates: 4240
  New cells created: 4028
  Cells with invalid positions: 212
  Cell updates: 0
```

10
Data Management

Hierarchy Management

Loading Hierarchies – loadHier

The loadHier utility can be used to load and refresh a hierarchy. All hierarchy data files are saved in fixed width (or space delimited) files with a ".dat" file extension. The width of positions is set by a configuration file before a domain is built. The width of positions can be increased after a domain has been built using the **inithier** utility.

RPAS allows for multiple input files to be loaded for the same hierarchy. The primary use for this is loading hierarchy data for multiple languages. Hierarchy files are in the format **hier.dat.***, where **hier** is name of a registered hierarchy. The "**.***" indicates that all extensions of a hierarchy file are loaded for a given hierarchy. For example, a run of the **loadhier** utility would load **prod.dat, prod.dat.japanese,** and **prod.dat.spanish** if those files were available in the input folder.

RPAS automatically generates a backup copy of hierarchy files prior to performing a load for a hierarchy. If any type of error occurs during the load process the hierarchy is restored from the backup copy.

Usage

loadHier –version

loadHier –d *domainPath* –load *hiername* {–purgeAge *purgeage*} {–checkParents} {–noClean}{–loglevel *level*}

Argument	Description
–version	Use this argument to get the version information. It does not require –d domainPath.
–d *domainPath*	Specifies the domain in which to load the hierarchy.
–load *hierName*	Specifies the name of the hierarchy to load and refresh.
–purgeAge *purgeage*	Specifies the purgeage during loadHier. If not specified, loadHier gets purgeage from domain.
–checkParents	Use this argument to check the parents while loading.
–noClean	If specified input files and the meta data used during load process are not cleaned. It is used only for debugging purposes.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Updating Domains with Hierarchy Changes – reshapeArrays

The **reshapeArrays** utility is used to make arrays conform to the current hierarchies in the domain. Any positions added to dimensions as a result of running loadHier will be added to arrays that have this dimension. Any positions removed from dimensions will be removed from the arrays that have this dimension. **reshapeArrays** will update the arrays to reflect these changes made in the hierarchies.

Usage

reshapeArrays –version

reshapeArrays –d *domainPath* –registered {–purge} {–processes *max*}

reshapeArrays –d *domainPath* –db *dbName* {–array *arrayName*} {–purge}

Argument	Description
–version	Use this argument to get the version information.
–d *domainPath*	Specifies the domain containing the arrays to be reshaped.
–registered	Use this argument to reshape all registered arrays. You must specify either –registered or –db, but not both.
–processes *max*	In a Global Domain environment, this will run the utility with the specified number of parallel processes on the master domain and local domains. If this parameter is used with the –db option or in a simple domain, –processes will be ignored.
–db *dbName*	If you specify –db, all arrays in the specified database are reshaped. You must specify either –registered or–db, but not both. dbName can specify either a full or relative path to the database. If the path is relative, it is relative to the root of the domain.
–array *arrayName*	Specifies which arrays in a database are to be reshaped. This argument can be repeated to reshape multiple arrays. If –array is not specified, all arrays in the specified database are reshaped. –array can be repeated on the same command line to reshape multiple arrays.
–purge	If –purge is not specified, the original database will be copied to dbName.bak.

Reconfiguring Global Domain Partitions – reconfigGlobalDomainPartitions

It is common for many customers to regularly add, remove, or change the parent-child relationships for positions in hierarchies, most commonly for positions in the product hierarchy. While this movement/reassignment of positions is normally handled with the loadhier utility, a special process must be followed for positions at the partition level of a Global Domain environment.

The following information is only relevant if an environment has been built as a Global Domain.

Consider a Global Domain environment that is partitioned at the "department" dimension/level in the product hierarchy. This environment could handle the addition or removal of SKU's, classes, or divisions without having to follow a special process. However, if a new department is to be added to the environment, an administrator must run a special RPAS utility to add this position before it can be loaded into the hierarchy.

The following process must be followed to add or remove a position at the partition level of a Global Domain environment:

- Administrator must be notified in advance that a new position is being added to or removed from the system
- Run the utility **reconfigGlobalDomainPartitions** to add/remove the positions by specifying the local domain to which the positions do or will belong
- Load the updated hierarchy file (that contains the new positions or doesn't contain the removed positions) using the **loadhier** utility; the next run of **loadhier** following the addition or removal of positions using the **reconfigGlobalDomainPartitions** utility must contain the updated hierarchy file or **loadhier** will fail.
- Run **reshapearrays** utility to update the domain to reflect the changed positions.

> **Note:** This only applies to positions at the partition level. Positions can be added or removed below or above the partition level simply by being added or removed from the hierarchy file.

Usage

reconfigGlobalDomainPartitions –d *pathToMasterDomain* –add *posName1,posName2, ...* –sub *pathToSubDomain*

reconfigGlobalDomainPartitions –d *pathToMasterDomain* –remove *posName1, posName2, ...*

Argument	Description
–d *pathToMasterDomain*	Specifies the path to the master domain in a Global Domain environment.
–add *posName1, posName2,*	Adds one or more positions to a specified local domain. The path to the local domain must follow the list of positions to add.
–remove *posName1, posName2, ...*	Removes the designated positions from the local domain to which the positions belong.
–sub *pathToSubDomain*	Specifies the path to the local domain to which new positions are being added.

Notes:

Position names are separated by commas and must be valid external position names without the prefix of a dimension.

The path to the local domain does not need to be specified if removing positions.

Assumptions and Limitations

Positions can only be added or removed in EXISTING local domains.

The user has to specify the local domain to which the positions are being added.

The user must call this utility before loading the hierarchy.

The user must run reshapeArrays utility after loading the hierarchy.

The remove option will NOT remove the local domain if local domain does not have any positions.

Multiple positions can be added to a local domain in a single call to the utility.

If new positions are included in the hierarchy load file prior to running the loadhier utility, RPAS will reject the entire load for that hierarchy file, generate an error message, and use the previous load.

Defining Hierarchy Settings – inithier

The inithier utility should be used for the following activities:

- Turning on or off position-level security. This can also be controlled in the Configuration Tools.
- Changing the width of positions for a dimension. The default value is 24. This value can be increased if necessary.

For information about position-level security, refer to the write-up on page 15. Note that position-level security can also be established in the RPAS Configuration Tools.

Position-level security must be set for a specific dimension of a hierarchy. Use the following command to enable or disable this setting:

inithier –d *pathToDomain* –h *hierarchyName* (–securityDim *dimName* | –noSecurity) {–loglevel *level* }

- The other use of the **inithier** utility is to extend the length of position names for a dimension to a value greater than the default value of 24 characters. RPAS 11.1 provides the ability for any dimension to have position names longer than 24 characters.

  To extend the length of position names use the following command:

  inithier –d pathToDomain –dim dimensionName –width width

  Note: The designated length of the position names must correctly match the associated data of those positions names in the input data files.

  Note: The length can only be increased and cannot be decreased after a chance has been made.

The list of all arguments for **inithier** is provided below.

Argument	Description
–version	Use this argument to get the version information.
–d *domainPath*	Specifies the path to the domain.
–h *hierarchyName*	Specifies the name of the hierarchy for which position-level security is enabled or disabled. Use –h with either –securityDim or –noSecurity.
–securityDim *dimName*	Specifies the name of the dimension for which position-level security is enabled.
–noSecurity	Use this argument to disable position-level security for the hierarchy specified using the –h argument.
–dim *dimensionName*	Used to specify the dimension for which the width of position names is to be increased.
–width *width*	Specified the new width of positions for the specified hierarchy. This must be an integer value greater than the previous setting (default is 24 characters).

Argument	Description
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Renaming positions – renamePositions

- RPAS provides the ability to change the name of a position using an RPAS utility named **renamePositions**. Positions that are to be renamed should be included in a hierarchy data file that is located in a designated input directory (specified when running utility) and that follows the naming convention **hierName.rn.dat** (for instance, “prod.rn.dat”).
- After the hierarchy data file(s) has been updated and placed in the designated location, an administrator must run the **renamePositions** utility with the following usage:

  renamePositions –d *domainPath* –i *inputDirectory* –hier *hierName* {–log *logFileName*} {–n}

Argument	Description
–d *domainPath*	Specifies the full path to the domain
–i *inputDirectory*	Input directory where input file with positions to rename is located. Utility looks for hierarchy data files with “rn” between hierarchy name and .dat extension (for instance, prod.rn.dat).
–hier *hierName*	Hierarchy for which positions are being renamed.
–log *logFileName*	Optional parameter to generate log file to file name other than default (default file name is hierRename.log).
–n	Do not apply changes, but generate a report that identifies changes that would be applied

Note the following about the input file:

- There will be three columns of data in each input line corresponding to dimension name, old position name, and new position name. The three fields will be tab-delimited. Any line not formatted this way will be ignored. Empty lines are also ignored.
- Old position names must be an existing position name.
- New position names cannot be an external name that is already in use or an existing internal name. Lines having invalid position names will be ignored and added to the log file.
- Old Position Name and New Position Name should not be prefixed with the name of the dimension.
- The 'width' attribute in the domain must be greater than or equal to the maximum length of the new external names in the input file. Otherwise, width reconfiguration must be done before the rename process (using the inithier utility). If the width of the new name is greater than the width attribute of the corresponding dimension, RPAS will print an error in the log file and ignore the record.
- Dimensions specified in input file should belong to the hierarchy specified in arguments. Otherwise, the record will be ignored and RPAS will print an error in the log file.

Note the following about the utilty:

- –n is a dry run, which means it does everything as a true run (that is, writing to a log file) except that it does not actually commit the changes to the domain.
- –log is an optional argument that you can use to name the log file other than the default, which will be created as hierRename.log in the current directory.

Measure Data

Loading Measure Data – loadMeasure

The **loadMeasure** utility is used to load data into the domain. You must specify the measure name and the path to the domain containing the measure.

Usage

loadmeasure –d *pathToDomain* –measure *measureName* {–logdirectory *directoryName*} {–applyloads} {–loglevel *level*}

Argument	Description
–d *pathToDomain*	Specifies the domain in which to load the measure.
–measure *measureName*	Specifies the name of the measure to load. The name must be lowercase.
–logdirectory *directoryName*	Specifies the location of the output error log. The default location is pathToDomain/scripts/err.
–applyloads	Use this argument to apply any staged loads for the named measure.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Exporting Measure Data – exportData

Use exportData to export measure data from RPAS into text files. Each line exported contains the position name for the exported dimension followed by the value in the cell for each array being exported. Note that more than one array may be exported and more than one dimension in each array can be exported.

The utility may be invoked by specifying all parameters on the command line or by specifying an array that contains a list of the parameters.

The parameters specify what arrays and dimensions are exported and how to format the data. It is best to specify the arrays first. An array specification begins with –array followed by the array information. This includes the array name, formatting string, NA cell value, and NA cell value formatting string. The formatting string for both the cell value and NA value is based on the C language printf function formats. See the documentation on the printf function if you are not familiar with the possible values. The –array parameter can be repeated as needed to export more than one array into the same export file. Remember that the order the arrays appear in the –array parameter is the order they will appear in the export file.

After the arrays have been specified, the next thing to do is specify the dimensions to be exported within the arrays. The –dim parameter is used to specify a dimension in an array. The –dim parameter is followed by the dimension name, a convert option, the formatting string (just like an array), and the order the dimension appears in the export file. Because arrays are not required to contain identical dimensions, it is important to list all dimensions in all arrays with the –dim parameter. This makes it possible to track dimensions across arrays and line the data up correctly. If a dimension in an array is not to be in the export file, set the last value of this parameter to 0. The conversion option specifies either the number of characters to be removed from the position name or it specifies an array that contains the real position name. If an array name is given, this array must be a vector. The function will go to this array and use the original position name to jump to the cell of the same position name. It will then get the cell value and use that as the position name in the export.

The –wide parameter causes the data to be exported wide meaning that the innermost dimension will go across the row instead of each cell on a separate line. This is most useful when the innermost dimension is time. This format puts all time data on one row of output with breaks for each of the other exported dimensions. The –range parameter can be used in conjunction with wide format (–wide) to specify a range along the innermost dimension. Only values in the range will be considered for export.

The –skipNA parameter is used for the NA suppression option. This option controls whether data is exported based on it having NA's in the export row. The choices for the –skipNA parameter are ANYNA, ALLNA, ALWAYS, or ARRAYNA. The ANYNA value will cause a row of data not to be exported if any cell contains a NA value. The ALLNA value will cause a row of data not to be exported only if all values in the row are NA. The ALWAYS value will export data regardless of whether it contains NAs. The ARRAYNA value will cause a row of data to not be exported if the value in the given array name is NA. If the ARRAYNA value is set then the –naArray parameter must be given. This parameter is followed by the array name to be used with the ARRAYNA value.

If all parameters are contained in an array, and then after the export file name and source database name, the -–params parameter is used to specify the database name and array name that contains all the parameters needed for the export.

Usage

exportData –version

exportData –d *domainPath* –out *outputFile* –params *db array*

exportData –d *domainPath* –out *outputFile* –meas \"*measSpec*\" {–wb *wbName*} {options}

exportData –d *domainPath* –out *outputFile* –array \"*arraySpec*\" {options}

Argument	Description
–version	Prints the RPAS version, revision, and build information.
–d *domainPath*	Specifies the domain containing the data you wish to export.
–out *outputFile*	Specifies the file that will contain the exported data. The outputFile is relative to the domain unless the full path is specified.
–params *db array*	Instead of specifying all parameters on the command line, this parameter allows the parameters to be read from an array. db specifies the name of a Gem file where the array of parameters is stored. array specifies the name of an array in the specified database that has the above parameters
–meas	Specifies the measure to export. measSpec must be quoted, and the format is \"measName cellFormat naValue naFormat\". The –meas argument can be repeated to export multiple measure arrays to the same output file. Measures are exported at the base intersection.

Argument	Description
–array	Specifies the array to export. arraySpec must be quoted, and the format is \"dbName|arrayName cellFormat naValue naFormat\". dbName can be a path to the database (relative paths are relative to the domain root). Both cellFormat and naFormat use printf format commands. See the documentation on the printf function if you are not familiar with the possible values. The –array argument can be repeated to export multiple arrays to the same output file. The order in which arrays are listed is the order in which they will be exported.

Option	Description
–dim \"*dimSpec*\"	Specifies the dimension to be exported. dimSpec must be quoted, and the format is \"dimName *conversion format order*\". *conversion* is either a count of the number of characters to strip from the start of the position name or the name of an array to be used to translate the position name before writing to the output file. *format* is a printf-style format for the position names. Please see the documentation on the printf function if you are not familiar with the possible values. *order* indicates the order the dimension is listed in the output file. If the value is 0, then the dimension is not exported. The –dim parameter can be repeated.
–skipNA always|allna|anyna|arrayna	Controls whether a line of data is exported based on having NAs in a cell. *always* exports data regardless of NAs. *allna* does not export if all columns are NA (default). *anyna* does not export if any column is NA. *arrayna* does not export if cell value in an array is NA (requires –naArray).
–naArray *arrayName*	When *arrayna* is specified using the –skipNA parameter, this option specifies the export array that is checked to determine if data is exported.

Option	Description
–index *arrayName*	Controls whether arrays are indexed by looking at a specified array. Only export the non-NA cells in the given array and each cell in the other arrays that have the same position names. If another array is at a higher dimension level, translate the given arrays cell index to the other arrays.
–range *start:end*	Used in conjunction with the –wide parameter, specifies a range along the innermost dimension. Only values in the range are considered for export.
–append	Specifies that output is appended at end of output file. The default is to overwrite output file.
–wide	Exports data in innermost dimension across instead of down.
–time	Specifies a YYYYMMDD format for dates.

Mapping Data between Domains – mapData

- The **mapData** utility is used to move data from one domain to another. Specifically, it copies data from an existing domain, database, or array to a new domain, database or array.
- Before running this utility, the new hierarchy must be loaded in the destination domain. After **mapData** has copied data, administrators can purge the source domain by calling **loadHier** with a purge age of 0. Tasks such as hierarchy loading, hierarchy purging, and the validation of source and destination domains will be performed outside of this utility.

Usage

mapData -srcDomain *srcDomainPath* -destDomain *destDomainPath* {-db *dbName* {-array arrayName}} {-loglevel}

Argument	Description
-srcDomain *srcDomainPath*	Specifies the full path of the source domain.
-destDomain *destDomainPath*	Specifies the full path of the destination domain.
-db *dbName*	Instructs **mapData** to map data only on the given database.
-array *arrayName*	Instructs **mapData** to map data only on the given array within the given database specified by **–db**.
-loglevel	Sets the logger verbosity level.

Run **mapData** without the **-db** argument to map the entire domain.

To set the logger verbosity level, use **-loglevel**. Possible values include **all**, **profile**, **debug**, **information**, **warning**, **error**, or **none**.

Moving Data between Arrays – updateArray

The updateArray utility moves data from a source array to a destination array. The destination array must contain the superset of dimensions in both source arrays. The source array's dimensions may be at the same or higher level (as mapped by the dimension dictionary). If a dimension in the source array is at a higher level, the results are spread across the lower level dimension in the destination. If there are extra dimensions in the destination array, the results are replicated across these extra dimensions. The NA value of the destination array remains unchanged.

To limit the scope of the update, a mask array and an innermost range may be specified. If a mask array is given, the update is limited to cells in the source array for which the corresponding mask cell is on. If an innermost range is given for source or destination array, the update is limited to cells that are within the start and end of this range on the innermost dimension. If the source and destination arrays are not in the same domain, the measure store associated with the source domain is used to find hierarchy information.

Usage

updateArray –destArray *dbPath.arrayName* {–srcArray *dbPath.arrayName*} {–destDomain *domainPath* {–srcDomain *domainPath*} {–maskDomain *domainPath*} {–maskArray dbPath.*arrayName*} {–updateMethod *method*} {–srcRange *first:last*} {–destRange *first:last*} {–srcScalar *scalarCell*} {–version} {–loglevel *level*}

OR

updateArray –argFile *filename* {–version} {–loglevel *level*}

Argument	Description
–destArray *dbPath.arrayName*	Required argument. Specifies the destination array. dbPath is relative to destDomain.
–srcArray *dbPath.arrayName*	Optional argument. Default is no source array. **Note:** This parameter cannot be used with –srcScalar scalarCell.
–destDomain *domainPath*	Optional argument. Default is current working directory.
–srcDomain *domainPath*	Optional argument. Default is current working directory.
–maskDomain *domainPath*	Optional argument. Default is current working directory.
–updateMethod *method*	Optional argument. Default is OVERLAY. The following update methods are available: SKIPNA – omit NA cells in source. SKIPPOP – omit populated cells in source. OVERLAYNA – update NA cells in destination. OVERLAYPOP – update populated cells in destination. OVERLAY – update all cells in destination with source.
–srcRange *first:last*	Optional argument. Default is no range. Defines range along innermost dimension of source array.

Argument	Description
–destRange *first:last*	Optional argument. Default is no range. Defines range along innermost dimension of destination array.
–srcScalar *scalarCell*	Optional argument. Default is NA cell. Format for scalar cell is one of: NUMERIC: numeric value STRING: literal value BOOL: Boolean value DATE: date value NA. **Note:** This parameter cannot be used with –srcArray dbPath.arrayName.
–loglevel *level*	Use this argument to set the logger verbosity level.Possible values: all, profile, debug, information, warning, error, or none.
–version	Use this argument to get the version information.

11

Miscellaneous RPAS Utilities

Operational Activities

Alerts

Overview

Alerts are an exception management tool for users. An alert is a measure that evaluates a business rule (returning a value of true or false). RPAS then notifies users of the “true” conditions and allows users to build workbooks to resolve the scenario that drove the alert.

Alert measures are first registered in the domain. These measures are of type Boolean, which means they have a value of true or false. Next, rules (expressions) are then registered in the domain for the alert measures to define the business rules used to evaluate the alert.

Once the registration process is complete, the alert utility is run to “find” the alerts in the domain. After the alert finder runs, the user can view the identified alerts in the Alert Manager in the RPAS client.

The following is a high level description of the process for defining an alert.

- Register an alert measure – This must be a Boolean measure (values are true-false, yes-no, and so on) and must be registered with the RPAS utility regmeasure or in the RPAS Configuration Tools.
- Register the alert and the expression for which the alert should be evaluated. This flags the registered measure as an alert so that it is recognized when the “alert finder” is run.
- The “alert finder” is run on the domain and determines when the value of the alert expression is true.

Registering Alert Measures – regmeasure

Alert measures can be registered via the RPAS Configuration Tools or the RPAS utility **regmeasure**. The type of this measure must be Boolean.

> **Note:** This utility is centralized in a Global Domain environment and can only be run on the master domain.

regmeasure –d pathtodomain –add alertMeasName –type boolean –baseint baseintersection –db databasename

Argument	Description
–d *pathdodomain*	Specifies the path to the domain where the alert measure is to be registered.
–add *alertMeasName*	Adds the new measure to the domain using the designated name. **Note:** This name must be less than or equal to 15 characters.
–type boolean	Registers the alerts as a Boolean type measure.
–baseint *baseintersection*	Specifies the base intersection of the measure.
–db *databasename*	Specifies the name of the database in which the measure will be stored.

Designate Alert and Define Alert Expression – alertmgr

Once the measure has been registered, the measure must be identified as an alert measure and must have an expression defined for which the alert condition will return a value of true or false.

This process is completed using the RPAS utility **alertmgr**. See the following section for the usage.

Find Alerts on the Domain – alertmgr

The “alert finder” is run in batch to evaluate all registered alert measures in the domain. This process is most commonly added to a batch process that is either run nightly or weekly.

alertmgr –d *pathtodomain* –findAlerts

alertmgr Utility

The following is the complete usage of the RPAS utility **alertmgr.** Use the **alertmgr** utility to perform the following:

- find alerts
- register and unregister alerts
- add expressions to alerts
- create and label alert categories

alertmgr –version

alertmgr –d *pathToDomain* –findAlerts {–loglevel *level*}

alertmgr –d *pathToDomain* –register *alertName* {–category *catName*} {–categoryLabel *label*} {–expression \"*expr*\"} {–loglevel *level*}

alertmgr –d *pathToDomain* –unregister *alertName* {–loglevel *level*}

alertmgr –d *pathToDomain* –addExpression *alertName* {–expression \"*expr*\"} {–loglevel *level*}

Argument	Description
–d *pathToDomain*	Specifies the directory in which to run the utility.
–version	Use this argument to get the version information. It does not require –d domainPath.
–findAlerts	Finds all the alerts in the specified domain.
–register *alertName*	Use this argument to register an alert. Requires –category, –categoryLabel, and –expression. **Important:** alertName must be the name of a registered Boolean measure.
–unregister *alertName*	Use this argument to unregister an alert. alertName must be the name of a registered Boolean measure.
–addExpression	Use this argument to add an expression to the alert. Use –expression to specify the expression.
–category *catName*	Use this argument to specify a category for the alert.
–categoryLabel *label*	Use this argument to specify a label for the alert category.
–expression \"*expr*\"	Use this in conjunction with –addExpression and –register. expr can be any valid RPAS 11 expression.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Copying Domains – copyDomain

The **copyDomain** utility is used to copy a simple domain or all domains included in a global domain implementation.

For a standard, simple (that is, non-global) domain, **copyDomain** copies the domain directory recursively from one location to another.

For a global domain environment, **copyDomain** copies the master domain to the specified destination, and then it copies each local domain into corresponding subdirectories of the new location. As part of this particular replication process, the utility updates all relevant arrays so that the domains are still connected together properly.

Relative paths are supported with this utility and are used when creating the new copies of all the underlying arrays. Relative paths are based on the full pathname of the domain's root directory.

Usage

copyDomain –d *pathToSrcDomain* -dest *pathToDest* { -f }

copyDomain –version

Argument	Description
–d *pathToSrcDomain*	Specifies the path of the domain to be copied.
–dest *pathToDest*	Specifies the path to where the domain is to be copied.
–f	Forces the deletion of the existing destination path before copying.

Setting Miscellaneous Domain Properties – domainprop

Use the domainprop utility to manipulate the properties of a domain. You can specify password properties, lock user accounts, and determine whether or not a daemon is currently managing a domain.

Usage

domainprop –version

domainprop –d *pathToDomain* –expirePassword {*days*} {–passwordHistory {*oldPasswordCount*}} {–property *propertyname=value*} {–lockAccount {*failedLogins*}} {–daemonPort}

Argument	Description
–d *pathToDomain*	Specifies the domain you want to manipulate.
–version	Use this argument to get the version information. It does not require –d domainPath.
–expirePassword {*days*}	Used to set or view the number of days a password is valid. If a number follows the argument, it sets the number of days a password is valid for. Otherwise, it prints the current setting.
–passwordHistory {*oldPasswordCount*}	Used to set or view the number of previous passwords that are kept to ensure that a user does not repeat his password too often. If a number follows –passwordHistory, the property is set to that number. Otherwise, the current setting is printed.
–property *propertyname=value*	Used to specify the property to be changed. See the list of properties below that can be set with this utility.
–lockAccount {*failedLogins*}	Used to set or view the number of failed login attempts that can occur before the account is locked out. If a number follows –lockAccount, the property is set to this value. Otherwise, the current setting is printed.

Argument	Description
–daemonPort	Prints a message indicating whether or not the domain is currently being managed by a DomainDaemon. If the domain is not currently being managed, the port of the last daemon to manage the domain is printed.

Available properties

disable_commit_later (value is Boolean) – setting this property to "true" (or "t") disables the ability to use "Commit Later" in the File menu of the RPAS client; this property is set to false by default.

Calculation Engine – mace

The mace utility (Multi-dimensional Array Calculation Engine) allows you to evaluate rule groups or expressions in order to manipulate measures. This allows the use of the RPAS 11 calculation engine in batch. The most commonly used commands for batch processes will be:

mace –d *domainPath* –run –group *groupName*

OR

mace –d *domainPath* –run –expression "*temporary expression*"

The first will cause an entire rule group to be evaluated. The second will evaluate a single expression.

The mace utility can also be used to:

- create rules and rule groups
- add rules to rule groups
- add expressions to rules
- delete rules not contained in a rule group
- remove any or all rule groups
- validate expressions
- print a list of rules or rule groups

Usage

mace –version

mace –d *domainPath* –find *string*

mace –d *domainPath* –newGroup *groupName*

mace –d *domainPath* –newRule

mace –d *domainPath* –addRule *groupName:ruleName*

mace –d *domainPath* –addExpression *ruleName* –expression *exprString*

mace –d *domainPath* –check –expression *expString*

mace –d *domainPath* –run (–group *groupName* | –expression *expString*) {–debugRuleEngine}

mace –d *domainPath* –resolve *groupName* –measures *measureList* {–debugRuleEngine}

mace –d *domainPath* –transit *workbookName* –group *groupList* {–debugRuleEngine}

mace –d *domainPath* –print (–rule *ruleList* | –group *groupList* | allGroups)

mace –d *domainPath* –removeGroup *ruleGroup*

mace –d *domainPath* –purgeRules

mace –d *domainPath* –removeAllRuleData

mace –d *domainPath* –validate (general | calc) –ruleGroup *groupName*

mace –d *domainPath* –validate *refresh* –ruleGroup *groupName* –calcRuleGroup *calcGroupName*

Argument	Description
–version	Use this argument to get the version information. It does not require –d domainPath.
–d *domainPath*	Specifies the domain in which to load the measure.
–find *string*	Use this argument to search all expressions for the specified string, printing all the rules and rule groups that have these expressions.
–newGroup *groupName*	Use this argument to create a new rule group with the specified name.
–newRule	Use this argument to create a new empty rule.
–addRule *groupName:ruleName*	Use this argument to add the specified rule to the specified rule group.

Argument	Description
–addExpression *ruleName*	Use this argument to add an expression to the specified rule.
–expression *exprString*	Use the argument to specify the expression. This argument is used in conjunction with the –addExpression, –check, and –run arguments.
–check	Use this argument to validate the specified expression.
–run	Use this argument to evaluate the specified expression or rule group.
–group *groupName*	Use this argument to specify the rule group to evaluate using the –run argument.
–resolve *groupName*	Use this argument to order (does not evaluate) expressions within rule group. Requires a comma-separated list of edited measures.
–measures *measureList*	Use this argument to specify the measures to resolve.
–transit *workbookName*	Use this argument to run a calculation engine by transmitting over a list of rule groups. Requires the name of an existing workbook and a comma-separated list of rule-group names.
–group *groupList*	Use this argument to specify a list of group names, separated by commas. Use this argument in conjunction with the –transit and –print arguments.
–print	Use this argument to print all the specified rules and rule groups. The ruleList is a comma-separated list of rule names. The groupList is a comma-separated list of group names. If "true" is supplied for either ruleList or groupList, all rules or rule groups are printed.

Argument	Description
–rule *ruleList*	Use this argument to specify a list of rule names, separated by commas. Use this argument in conjunction with the –print argument.
–allGroups	Use this argument in conjunction with the –print argument to print all rule groups.
–debugRuleEngine	Use this argument to generate a file "mace.log" in the working directory for logging RuleEngine specific debug information.
–addGroup	Use this argument to create a new rule group with the specified name.
–removeGroup	Use this argument to remove specified group and non-shared rules in it.
–purgeRules	Use this argument to remove all rules not contained in any rule groups.
–removeAllRuleData	Use this argument to remove all rule groups and all rules.
–validate	Use this argument to validate rule groups. If you need to validate a calc rule group, use the calc param. If you need to validate a refresh rule group, use the refresh param along with the –calcRuleGroup param to specify the corresponding calc rule group. For all other types of rule group, use the general param.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Managing Users – usermgr

Use the usermgr utility to add a user, remove a user, or print information about a user in a specified domain.

Usage

usermgr –version

usermgr –d *domainPath* –add *userName* –label *label* –password *psw* –group *grp* {–admin} {–loglevel *level*}

usermgr –d *domainPath* –remove *userName* {–loglevel *level*}

usermgr –d *domainPath* –list {–loglevel *level*}

usermgr –d *domainPath* –print –user *username* {–loglevel *level*}

usermgr –d *domainPath* –print –group *groupname* {–loglevel *level*}

Argument	Description
–version	Use this argument to get the version information. It does not require –d domainPath.
–d *domainPath*	Specifies the path to a domain that you want to add, remove, or get information about a user.
–add *userName*	Use this argument to add a user with a specified name. Use the other arguments specified in the usage to add those attributes for that user.
–label *label*	Use this argument to specify the label of the user that you are adding to the domain.
–password *psw*	Use this argument to specify the password of the user that you are adding to the domain.
–group *grp*	Use this argument to specify the group of the user that you are adding to the domain.
–admin	Use this argument to specify that the user you are adding to the domain has administrative rights.
–remove *userName*	Use this argument to remove the user with the specified name from the domain.
–list	Use this argument to list all the users registered to the specified domain.
–print	Use this argument to print the specified user or group information.
–user *username*	Use this argument to specify the user name in the specified domain that you want to print. This argument is only applicable to –print option.

Argument	Description
–group *groupname*	Use this argument to specify the group in the specified domain name that you want to print. This argument is only applicable to –print option.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Managing the Workbook Batch Queue – wbbatch

The **wbbatch** utility is used to:

- Manage workbooks in the **workbook batch queue**.
- Build workbooks that have been scheduled to be automatically built using the "Auto-Workbook Build" wizard in the RPAS client.
- Commit data from workbooks that have been scheduled for deferred commits using the "Commit Later" option in the RPAS client (File menu).
- Refresh workbooks that have been scheduled to be refreshed using the workbook's default refresh rule group.

Usage

wbbatch –d *pathToDomain* –build {*workbookName*} {–loglevel *level*}

wbbatch –d *pathToDomain* –refresh {*workbookName*}{–loglevel *level*}

wbbatch –d *pathToDomain* –scheduleRefresh *wbName* {–loglevel *level*}

wbbatch –d *pathToDomain* –unscheduleRefresh *wbName* {–loglevel *level*}

wbbatch –d *pathToDomain* –commit {*workbookName*}{–loglevel level}

wbbatch –d *pathToDomain* –scheduleCommit wbName {–loglevel *level*}

wbbatch –d *pathToDomain* –unscheduleCommit wbName {–loglevel *level*}

Argument	Description
–d *pathToDomain*	Specifies the domain containing the workbooks.
–version	Use this argument to get the version information. It does not require –d domainPath.
–build {*workbookName*}	Builds workbooks scheduled to be built automatically. To build a single workbook in the auto-workbook build queue, specify the name of the workbook. If no name is provided, all workbooks in the queue will be built.

Argument	Description
–commit {*workbookName*}	Commits workbooks with deferred commits. To commit a single workbook, in the commit later queue, specify the name of a workbook. If no name is provided, all workbooks in the commit later queue will be committed.
–refresh {*workbookName*}	Refreshes workbooks scheduled to be refreshed using this utility. To refresh a single workbook in the queue, specify the name of the workbook. If no name is provided, all workbooks scheduled to be refreshed will be completed.
–scheduleRefresh	Schedules a workbook to be refreshed later by adding it to the workbook batch queue.
–unscheduleRefresh	Removes a workbook from the workbook batch queue.
–scheduleCommit	Schedules a workbook to be committed later by adding it to the workbook batch queue.
–unscheduleCommit	Removes a workbook from the workbook batch queue.
–loglevel *level*	Use this argument to set the logger verbosity level.Possible values: all, profile, debug, information, warning, error, or none.

Workbook Manager – wbmgr

Use the wbmgr utility to inspect or remove the existing workbooks. Do not assume that manual removal of the workbook directories will remove the workbook metadata in the domain.

Usage

wbmgr –version

wbmgr –d *pathToDomain* –list –all {–loglevel *level*}

wbmgr –d *pathToDomain* –list –user *userName* {–loglevel *level*}

wbmgr –d *pathToDomain* –print –wbList *wb1,wb2,...* {–loglevel *level*}

wbmgr –d *pathToDomain* –remove –all {–loglevel *level*}

wbmgr –d *pathToDomain* –remove –user *userName* {–loglevel *level*}

wbmgr –d *pathToDomain* –remove –user *userName* –wbList *wb1,wb2,...* {–loglevel *level*}

Argument	Description
–version	Use this argument to get the version information. It does not require –d domainPath.
–d *pathToDomain*	Specifies the domain containing the workbooks.
–list –all	Lists all workbooks in the domain.
–list –user *userName*	Lists all workbooks belonging to the user.
–print –wbList *wb1,wb2,...*	Prints detailed information about workbooks in the list.
–remove –all	Removes all workbooks from the domain.
–remove –user *userName*	Removes all workbooks from the domain belonging to the specified user.
–remove –user *userName* –wbList *wb1,wb2*	Removes all the workbooks in the specified list for the specified user.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Informational RPAS Utilities

There are numerous RPAS utilities that can be used for finding information about many of the different components of a domain or domain data. The following utilities are for retrieving information and not to make any changes to a domain or data in a domain.

Checking the Validity of a Domain – checkDomain

This utility is used to check the validity of an existing domain. Currently, its primary purpose is to verify that a master domain matches its respective local domains. All discrepancies are written to the terminal.

checkDomain –d *pathToDomain* –type *expectedType* {-q}

Argument	Description
–d *pathToDomain*	Path to the domain that needs to be validated.
-type *expectedType*	expected type of domain: **simple**, **master**, or **sub**
-q	quiet mode - do not display progress messages

Valid domain types

simple: domain is a standard, non-partitioned domain

master: domain is the master of a global domain set

sub: local domain is one partition of a global domain set

When **checkDomain** is run on a **simple** domain (that is, non-Global Domain), the following two items get validated:

1. The domain directory exists
2. It is of type "simple"

If **checkDomain** is run on a **Global Domain**, it verifies that the global domain exists, that it is of type "master"; and it checks all of the sub-domains for:

1. The sub-domain directory exists and is of type "sub."
2. If the master domain has a repos directory, the sub-domain also has a repos directory.
3. The measures, rules, rule groups, templates, and functions are the same in the global and sub-domain.

If it is run on a **sub-domain**, it checks all of the items listed above for the global domain, but the validation is only performed between the global domain and the specified sub-domain.

Determining RPAS Server Version – rpasversion

Use the **rpasversion** utility to determine which version of the RPAS server is running in a particular location.

Usage

rpasversion –l *pathToLibrary*

List Contents of a Database – listDb

Use the listDb utility to list the contents of a database.

Usage

listDb –version

listDb –db *dbName* {–loglevel *level*}

Argument	Description
–version	Use this argument to get the version information.
–db *dbName*	Specifies the database for which you want to list the contents.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Printing Data from Arrays – printArray

Use **printArray** to print the contents of an array.

Usage

printArray –version

printArray –array *db.array* –specs {–maxpos *num*}

printArray –array *db.array* {–cell "*dim1:pos1,dim2:pos2,...*" {–format "*formatString*"}

printArray –array *db.array* –slice "*dim1:pos1,dim2:pos2,...*" {–format "*formatString*"} {–cellsperrow *num*} {–noposnames}

printArray –array *db.array* –allpopulatedcells {–format "*formatString*"} {–cellsperrow *num*} {–noposnames}

Argument	Description
–version	Prints the RPAS version, revision, and build information.
–array *db.array*	Specifies the array you want to print. You must specify the full path to the database containing the array. Do not specify the .gem suffix.
–specs	Prints the specifications of the array and positions along each dimension.
–cell	Identifies a single cell. Must not contain spaces. Specify using the format “dim1:pos1,dim2:pos2,…”
–slice	Identifies a single 1-D slice. Must not contain spaces. Specify using the format “dim1:pos1,dim2:pos2,…”
–allpopulatedcells	Print all populated cells including the navalue of the array.
–format	If –format is specified, any cells with numeric values are interpreted as dates. formatString determines how dates are interpreted, and can include: %Y - 4 digit year %m - month number (01 to 12) %d - numeric day of month (01 to 31) %H - 24 hour clock (00 to 23) %M - minute (00 to 59) %S - seconds (00 to 61) %s – milliseconds

Argument	Description
–maxpos	Used in conjunction with –specs, prints only the first *n* positions for each dimension.
–cellsperrow	Specifies how many cells should be printed in each row.
–noposnames	Suppresses the output of position names.

Printing Data from Measures – printMeasure

Use the printMeasure utility to print measure information.

Usage

printMeasure –version

printMeasure –d *pathToDomain* –m *measure* {–wb *workbookName*} {–specs} {–listDataIntersections} {–printData *aggType.intersection*} {–loglevel *level*}

printMeasure –list

Argument	Description
–version	Use this argument to get the version information. It does not require –d domainPath.
–d *pathToDomain*	Specifies the domain containing the measure you want to print.
–m *measure*	Specifies the measure you want to print.
–wb *workbookName*	Specifies the workbook associated with the measure you want to print. If a workbook is not specified, the domain measure information is printed.
–specs	
–listDataIntersections	
–list	It doesn't require –d domainPath.
–printData *aggType.intersection*	Prints out the nobs, nods and sparse format of the measure array at the specified intersection and agg type.
–loglevel *level*	Use this argument to set the logger verbosity level. Possible values: all, profile, debug, information, warning, error, or none.

Retrieving Domain Information – domaininfo

The **domaininfo** utility provides basic information about a given domain. Specifically, it can identify whether a domain is a simple domain (not created in global domain environment), a master domain (parent of a global domain), or a local domain (child of global domain).

The domain path (**-d**) is required for all commands except **–expectedversion**.

The **-domainversion** command gives the version of the domain pointed to by the **-d** parameter. The **-expectedversion** command prints the domain version that the current code would expect to find.

The **-type** command returns which of three types of domain is being inspected:

- Simple: domain is a standard, non-partitioned domain
- Master: domain is the 'master' of a global domain set
- Sub: domain is one partition of a global domain set

The **-sparsity** command reports whether a domain has been designated as containing sparse or hypersparse measure databases.

If a Global domain is being inspected, the **-listsubdomains** and **-subdomain** commands may be useful. The **-listsubdomains** command will display a list of all subdomains of this domain, and it will tell which positions of the partitioned dimension are in each subdomain. The **-subdomain** command allows searching for a particular position of the partitioned dimension or any dimension that rolls up to the partitioned dimension. For example, if the global domain environment is partitioned on DEPT, one may query which local domain contains a particular DEPT position or a particular SKU position.

The **-history** command lists the different versions and upgrade actions taken on the domain.

The **-all** command reports all relevant information on the domain.

Usage

domaininfo –d *pathToDomain* [*commands listed below*]

domaininfo –expectedversion

domaininfo –version

Argument	Description
–domainversion	Display version of referenced domain.
–type	Display domain type (see types above).
–xnames	
–sparsity	Display domain sparsity (see types above).
–listsubdomains	List all subdomains of a global domain.
–subdomain *dim,pos*	Find subdomain containing given position.
–history	Display version history of domain.
–all	Display combined details of domain.

Argument	Description
–expectedversion	Display domain version expected by this code.
–version	Display version of this utility.

A

Appendix: Integration Guide

RDF and Merchandise Financial Planning Integration with RMS

Summary of Integration Approach with RMS

The strategy for the extraction of foundation data from RMS is for the extract programs (RMSE) to provide flat files in a generic format. For each solution that will use this data, transformation scripts are used to reformat the data as needed to produce a file suitable for loading into the application. For the instances of data coming from RPAS to non-RPAS applications, extract programs are specific to the application in need of the data. Other scripting languages are then used (Perl or AWK) to perform additional data formatting.

This document summarizes the following:

- RMS 11.0/RMS 10.0 to RDF 11.1 and Merchandise Financial Planning 11.1 transformation programs
  - Merchandise Hierarchy
  - Organization Hierarchy
  - Calendar Hierarchy
  - Store Close Dates
  - Store Open Dates
  - Daily Sales and Issues
  - Weekly Sales and Issues
  - Out of Stock Indicator
- RDF 11.1 to RMS 11.0/RMS 10.0 extract programs
  - Approved Forecasts and Standard Deviations (Cumulative Intervals)
- Grade (RPAS 11.1) to RMS 11.0/RMS 10.0 extract programs
  - Store Grades
- Curve (RPAS 11.1) to RMS 11.0/RMS 10.0 extract programs
  - Differentiator Profiles
- Merchandise Financial Planning 11.1 to RMS 11.0/RMS 10.0 extract programs

Environment Variable Setup

In addition to any variables identified in the RMS integration documentation, the transformation and/or extract programs require the following environment variables:

- $RPAS_INTEGRATION_HOME: Identifies the location of the integration scripts when /common/header.ksh is run. This variable is used for all integration scripts packaged with the ARPOPlatform EXCEPT those included in 'rfx' (see $RDF_HOME below).
- $TO_RPAS: The staging area for the data to be loaded into RPAS. This directory should be located at the same level as the root of the RPAS domain. For example, if the domain RDF is located in Domains directory (example: /Domains/RDF), then $TO_RPAS should located at the same level as RDF (example: /Domains/to_rpas).
- $FROM_RPAS: The staging area for the data extract out of RPAS. This directory should be located at the same level as the root of the RPAS domain. For example, if the domain RDF is located in Domains directory (example: /Domains/RDF), then $FROM_RPAS should located at the same level as RDF (example: /Domains/from_rpas).
- $RDF_HOME: Identifies the location of the root of the 'rfx' directory. The 'rfx' directory packaged with the ARPOPlatform should be added to the location 'rfx' directory packaged with the RMS RETL programs.
- $ RI_RMSVERSION: Identifies the RMS version for integration. Set to 11 for RMS 11.0, set to 10 for RMS 10.0. In case this variable is not set, the version defaults to RMS 10.0 (example: 11)

RDF and Merchandise Financial Planning Transformation Programs

Common Program for All Transformations

The **rdft.ksh** script runs all of the necessary data extraction and transformation scripts (**rmse_*.ksh** and **rdft_*.ksh**, respectively) that are needed to produce the files to be loaded into RPAS/RDF/Planning. Most of these scripts are run in parallel (as background jobs).

Usage: rdft.ksh [-x] [-c] [-d dir]

-x: This option will cause the execution of the RMS data extraction wrapper (rmse.ksh) to be skipped.

-c: This option will cause FILE_DATE in rmse_config.env to be set to the current date instead of using VDATE.

-d: This option will cause all programs executed by rdft.ksh to be obtained from the "dir" directory.

Transformations of Merchandise Hierarchy Data

rdft_merchhier.ksh is the primary script used to build the data for RPAS from the RMS Merchandise Hierarchy tables. The schema used to produce the output file depends on the attributes and differentiator settings in RMS:

- Case 1: If PROD_ATTRIBUTES_ACTIVE = False and DIFFS_ACTIVE = False, then rdft_merchhier.base.schema will be used to produce the file. In this case attributes and diff fields will not be included in the merchandise hierarchy file.
- Case 2: If PROD_ATTRIBUTES_ACTIVE = True and DIFFS_ACTIVE = False, then rdft_merchhier.attributes.schema will be used to produce the file. This schema must be manually edited to support a specific attribute model and must be kept in sync with rmse_attributes.schema and rmse_attributes.ksh (see the RMSE end user documentation).
- Case 3: If PROD_ATTRIBUTES_ACTIVE = False and DIFFS_ACTIVE = True, then rdft_merchhier.schema will be used to produce the file. In this case diff fields will be included in the merchandise hierarchy file.
- Case 4: If PROD_ATTRIBUTES_ACTIVE = True and DIFFS_ACTIVE = True, then an error will result. In this release, the combination of diffs and attributes is not supported.

Intermediate schema and scripts which may be used (depending on configuration options) to produce the merchandise hierarchy file:

rdft_diff.domain.schema

rdft_merchdiff.domain.schema

rdft_merchhier_diff_trans.ksh

rdft_merchhier_split_by_domain.ksh

rdft_clean_partition.ksh

Additional merchandise hierarchy support for issue domains is provided in **rdft_item_loc.ksh**. This script is designed to produce a full item list for issues domains, only containing items that exist in the warehouses.

> **Note:** Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Location Hierarchy Data

rdft_orghier.ksh is the primary script used to build the location data file needed for RPAS from the RMS Organizational Hierarchy Table.

The following five constants may be modified in the script based on location hierarchy data requirements:

- **COMPANY_NAME** – The label for the company position to be populated in the file.
- **COMPANY_ID** – The name for the company position to be populated in the file.
- **STORE_CLASS_CONCAT** – When set to "True", causes the STORE_CLASS to be concatenated on the left of the STORE_CLASS_DESCRIPTION field in the final Store data output file.
- **ADD_AT_SIGN_TO_WH_DESC** – When set to "True", will cause the WHSE_NAME field in the Warehouse ouput file to have an "@" prefix.
- **LONG_WAREHOUSE_RECORDS** – when set to "True", the Warehouse output records will consist of 16 fields. If it is "False", the records will contain only four fields, WH, WHSE_NAME, COMPANY and CO_NAME.

Intermediate schemas which may be used (depending on configuration options) to produce the location hierarchy file:

- rdft_issues.schema
- rdft_issues_long.schema
- rdft_orghier_store.schema

> **Note:** Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Calendar Hierarchy Data

rdft_calhier.ksh transforms the Calendar Hierarchy data extracted from RMS for loading into RPAS.

Configuration inputs to the script include:

DATE_PREF – The path to the file that contains text indicating whether the format of the Date Description field will be mm/dd/yyyy or dd/mm/yyyy. See the RMS Operations Guide for date format options.

LAST_DOW – The path to the file that contains a day of week name or abbreviation indicating which day of the week is considered to be the end of the week for the fiscal calendar being used at this installation.

Transformations of Daily Sales and Issues Data

rdft_daily_sales.ksh produces the daily sales and issues data files based on regular, promotion, clearance, and issues.

The following constant may be modified in the script based on data requirements:

- **DOM_START_COL** – Defines the starting column position of the Domain ID in the RETL output schema. This is needed by **rdft_merchhier_split_by_domain.ksh** to split the files by domain ID. If the OUTPUT_SCHEMA file is modified, the value of DOM_START_COL may also require modification from the default value.

Intermediate schemas which may be used (depending on configuration options) to produce the sales and/or issues data file:

- rdft_daily_sales.schema

> **Note:** Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Weekly Sales and Issues Data

rdft_weekly_sales.ksh produces the weekly sales and issues data files based on regular, promotion, clearance, and issues.

The following constant may be modified in the script based on data requirements

- **DOM_START_COL** – Defines the starting column position of the Domain ID in the RETL output schema. This is needed by rdft_merchhier_split_by_domain.ksh to split the files by domain ID. If the OUTPUT_SCHEMA file is modified, the value of DOM_START_COL may also require modification from the default value.

Intermediate schemas which may be used (depending on configuration options) to produce the sales and/or issues data files:

- rdft_weekly_sales.schema

> **Note:** Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Store Open Date Data

rdft_open_date.ksh produces the Store/Warehouse Opening Date data file.

Intermediate schema used to produce the store open date data files:

- rdft_open_date.schema

Transformations of Store Close Date Data

rdft_close_date.ksh produces the Store/Warehouse Closing Date data file.

Intermediate schema used to produce the store closing date data files:

- rdft_close_date.schema

Transformations of Out of Stock Indicator Data

rdft_outofstock.ksh produces the Store and Warehouse (issues) Out of Stock Indicator data extracted from RMS.

Intermediate schema and scripts which may be used (depending on configuration options) to produce the Out of Stock Indicator data file:

- rdft_outstock_split_by_domain.awk
- rdft_outofstock.schema
- rdft_outofstock_issues.schema
- rdft_outofstock_sales.schema

RDF and Merchandise Financial Planning Transformation Matrix

The following matrix identifies the transformation scripts and schemas used for each the hierarchy and data files produced for RDF 11.1 and Merchandise Financial Planning 11.1:

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out of Stock Indicator	Store Open Dates	Store Close Dates
rfx/lib	rdft_merchhier_diff_trans.ksh	X							
	rdft_merchhier_split_by_domain.ksh	X							
	rdft_outofstock_split_by_domain.ksh						X		
rfx/schema	rdft_close_date.schema								X
	rdft_daily_sales.schema				X				
	rdft_diff.domain.schema	X							
	rdft_merchierdiff.domain.schema	X							
	rdft_merchier.attributes.schema	X							
	rdft_merchhier.base.schema	X							
	rdft_merchhier.domain.schema	X							

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out of Stock Indicator	Store Open Dates	Store Close Dates
	rdft_merchhier.schema	X							
	rdft_open_date.schema							X	
	rdft_orghier_issues.schema		X						
	rdft_orghier_issues_long.schema		X						
	rdft_orghier_strore.schema		X						
	rdft_outofstock.schema						X		
	rdft_outofstock_issues.schema						X		
	rdft_outofstock_sales.schema						X		
	rdft_weekly_sales.schema					X			
rfx/src	rdft_ksh	X	X	X	X	X	X	X	X
	rdft_calhier.ksh			X					
	rdft_clean_partition.ksh	X							
	rdft_close_date.ksh								X
	rdft_daily_sales.ksh				X				
	rdft_item_loc.ksh	X							
	rdft_merchhier.ksh	X							
	rdft_open_date.ksh							X	
	rdft_orghier.ksh		X						
	rdft_outofstock.ksh						X		

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out of Stock Indicator	Store Open Dates	Store Close Dates
	rdft_weekly_sales.ksh					X			

RDF and Merchandise Financial Planning Extract Programs

Common Programs for Extracts

config.ksh is a configuration directory that requires both the RMS version being integrated and the backup action to be defined.

The following OPTIONAL arguments are available:

- Name of the domain: Defaults to directory name
- Number of the domain: Defaults to the 2 last digits of the directory name
- Format of timestamp attached to logs and processed input files: Defaults to: (date +"%b%d%a%I%M%p") (example: Aug02Thu0111PM)
- Data Drop: Defaults to ../../to_rpas
- Data Export: Defaults to ../../from_rpas
- Log Drop: Defaults to ./logs
- Error Drop: Defaults to ./err
- Reclass Data: Defaults to ../reclass_data

functions.ksh

This script file contains ksh functions that are used by scripts in [DOM]/scripts. It should be sourced, not executed in order to preserve environment variables.

header.ksh

This script should be run at the beginning of any implementation-specific script to setup function libraries, environment, and platform-specific routines.

Extract of Forecast Data for RMS

rdf_e_rms.ksh extracts forecast demand value and standard deviation (cumulative interval) at both day and week aggregations from an RDF domain.

Arguments:

-t: <Domain Type> (S for sales, I for issues)

-w: <Data Width> ([7...18], defaults to 12)

-d: <Domain> (defaults to current directory)

-n: <Domain Number> (defaults to last two digits of domain)

Output files:

${RPAS_EXPORT}/d<s|i>demand.<Domain Number> (demand at day)

${RPAS_EXPORT}/w<s|i>demand.<Domain Number> (demand at week)

Output file data format:

Field	Start	Width	Format
Day | EOW Day	1	8	Alpha
Product ID	1	20	Alpha
Location ID	26	20	Alpha
Demand	46	12	Alpha
Std. Dev. Demand	68*	12*	Numeric (floating point, 4 dec. digits with decimal)

* Width of Demand and Std. Dev. Demand may be overridden with the -w parameter; stated values Demand width and Std. Dev. Demand start and width are based on default width of 12.

Note: the following must be defined in the shell environment prior to calling this script:

RPAS_HOME

RPAS_INTEGRATION_HOME

Load of Extracted Forecast Data and Standard Deviations to RMS

rmsl_forecast.ksh pulls the daily/weekly forecast items into RMS.

During the loading of each domain file the following steps are performed:

1. Truncate the partition in the RMS forecast table which corresponds to the domain ID.

   Note: Partition names should always be in the format: [tablename]_[domainID]

2. Append a domain field and insert the domain_id into each record.
3. Load the forecast data into the RMS forecast table.

Usage: rmsl_rpas_forecast.ksh daily | weekly

Intermediate schemas which may be used (depending on configuration options) to produce the forecast data files:

rmsl_forecast_daily.schema

rmsl_forecast_weekly.schema

Extract of Diff Profile Data for RMS

profile_e_alloc.ksh extracts Curve diff profiles for use by Allocation.

Arguments:

-p: <Profile Number>

-m: <Mask Measure> (Optional mask; only positions for which the mask value is non-NA will be exported.)

-w: <Data Width> ([7...18], defaults to 12)

-d: <Domain> (defaults to current directory)

-n: <Domain Number> (defaults to last two digits of domain)

Output file: ${RPAS_EXPORT}/d1<Product Level>.<Domain Number>, where Product Level is the Aggregation intersection's Prod dimension

Output file data format:

Field	Start	Width	Format
Product ID	1	25	Alpha
Location ID	26	20	Alpha
Diff ID (optional)	46	36	Alpha
Quantity	82	12*	Numeric (floating point, 4 decimal digits, no decimal) * *Quantity width may be overridden with the -w parameter

Note: The following must be defined in the shell environment prior to calling this script:

RPAS_HOME

RPAS_INTEGRATION_HOME

Extract of Store Grade Data for RMS

grade_e_rms.ksh extracts store grades for use by RMS.

Arguments:

-t <Timestamp> (YYMMDDTTTT). This value corresponds to the timestamp of the Cluster Membership measure (clpm+<Timestamp>) to be extracted

-d <Domain> (defaults to current directory)

-n <Domain Number> (defaults to last two digits of domain)

Output file: ${RPAS_EXPORT}/gr<Timestamp>.<Domain Number>

Output file data format:

(a thru c) constitutes the header records

(d thru j) constitutes the detail records

(k thu l) constitutes the footer records

a. 'FHEAD'
b. Line ID Number
c. 'GRADU'
d. 'FDETL' (Record Identifier)
e. Line Sequence Identifier
f. Grade Group ID Number: This value corresponds to the first 8 characters of the Cluster Run Name measure (clnam+<user-defined name>) set by the user in the Generate Cluster wizard in Grade. For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
g. Grade Group: This value corresponds to the first 20 characters of the Cluster Run Name measure (clnam+<user-defined name>) set by the user in the Generate Cluster wizard in Grade. For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
h. Store ID
i. Grade Member Name
j. 'FTAIL' (Record Identifier)
k. Line ID Number
l. FDETL Line Total Number

Extract of Receipt Plan for RMS

plan_e_alloc.ksh extracts Merchandise Financial Planning measures for use by RMS

Arguments:

-m: <Measure Name>

-f: <Dif Dimension>

-w: <Data Width>] ([7...18], defaults to 12)

-d: <Domain> (defaults to current directory)

-n: <Domain Number> (defaults to last two digits of domain)

Output file: ${RPAS_EXPORT}/p1<Prod Dimension>.<Domain Number>

Output file data format:

Field	Start	Width	Format
Product ID	1	25	Alpha
Location ID	26	20	Alpha
Diff ID	46	36	Alpha (For future use; blank now)
EOW Date	82	8	Alpha
Quantity decimal)	90	12*	Numeric (floating point, 4 decimal digits, no

* Quantity width may be overridden with the -w parameter

Note: the following must be defined in the shell environment prior to calling this script:

RPAS_HOME

RPAS_INTEGRATION_HOME

RDF and Merchandise Financial Planning Extract Matrix

The following matrix identifies the extract scripts and schemas used for each the data files produced for RMS:

Directory	Script or Schema Name	Forecasts and Standard Deviations	Diff Profiles	Receipt Plan	Store Grades
common	config.ksh				
	functions.ksh	X			
	header.ksh	X	X	X	X
curve	profile_e_alloc.ksh		X		
grade	grade_e_rms.ksh				X
plan	Plan_e_alloc.ksh			X	
rdf	rdf_e_rms.ksh	X			
	rmsl_forecast.ksh	X			
	rmsl_forecast_daily.schema	X			
	rmsl_forecast_weekly.schema	X			