

Oracle® Retail Merchandising Suite
Customization and Extension Guide
Release 19.0
F27238-01

January 2020

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Primary Author:

Contributors:

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.

(iii) the software component known as **Access Via**TM licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(iv) the software component known as **Adobe Flex**TM licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

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

Contents

Send Us Your Comments	vii
Preface	ix
Documentation Accessibility	ix
Customer Support.....	ix
Improved Process for Oracle Retail Documentation Corrections	ix
Oracle Retail Documentation on the Oracle Help Center	ix
Conventions.....	x
1 Overview	1
2 Custom Flex Attributes	3
Defining Custom Flex Attributes.....	4
Group Sets.....	5
Groups.....	6
Managing Entities, Group Sets and Groups.....	6
Attributes	8
Custom Functions.....	11
Managing Record Groups.....	13
Using Custom Flex Attributes.....	15
3 In Context Launch.....	17
Merchandising.....	17
Allocation	20
Invoice Matching.....	21
Sales Audit	21
4 Custom Rules.....	25
Custom Merchandising Rules	25
Approval Rules	25
Custom VAT.....	29
Custom Conflict Checking Rules	30
Writing Custom Rule Functions	30
Configure Rule in Pricing	30
5 Application Customization	35
User Interface.....	35
Tasks	35
Reports Menu	41
Sidebar Menu	43
Application Navigator	47
Dashboards and Reports.....	49
Dashboards.....	49
Reports	52
Resource Bundles	57

Send Us Your Comments

Oracle Retail Merchandising Suite, Customization and Extension Guide, Release 19.0

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

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

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

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

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

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

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

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

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

Preface

The Oracle Retail Merchandising Customization and Extension Guide contains the requirements and procedures that are necessary for the retailer to extend and customize the Merchandising applications.

Documentation Accessibility

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

Access to Oracle Support

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

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL: <https://support.oracle.com>

When contacting Customer Support, please provide the following:

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

Improved Process for Oracle Retail Documentation Corrections

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

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

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

Oracle Retail Documentation on the Oracle Help Center

Oracle Retail product documentation is available on the following web site: <http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html>

Conventions

Navigate: This is a navigate statement. It tells you how to get to the start of the procedure and ends with a screen shot of the starting point and the statement “the Window Name window opens.”

This is a code sample

It is used to display examples of code

Note: In the images or examples below, user details / company name / address / email / telephone number represent a fictitious sample. Any similarity to actual persons, living or dead, is purely coincidental and not intended in any manner.

Overview

This document provides an overview of tools and options provided in the Merchandising Suite of applications for customizing or configuring the solutions to meet your business. The concepts in this document are applicable for both on premise and SaaS implementations of these solutions, unless otherwise noted. The solutions covered in the scope of this document include:

Functional Name	On Premise	Cloud Service
Merchandising	Oracle Retail Merchandising System (RMS)	Oracle Retail Merchandising Foundation Cloud Service - Merchandising module
Sales Audit	Oracle Retail Sales Audit (ReSA)	Oracle Retail Merchandising Foundation Cloud Service - Sales Audit module
Trade Management	Oracle Retail Trade Management (RTM)	Oracle Retail Merchandising Foundation Cloud Service - Trade Management module
Pricing	Oracle Retail Pricing (RP)	Oracle Retail Pricing Cloud Service (RPCS)
Allocation	Oracle Retail Allocation	Oracle Retail Allocation Cloud Service (RACS)
Invoice Matching	Oracle Retail Invoice Matching (ReIM)	Oracle Retail Invoice Matching Cloud Service (ReIMCS)

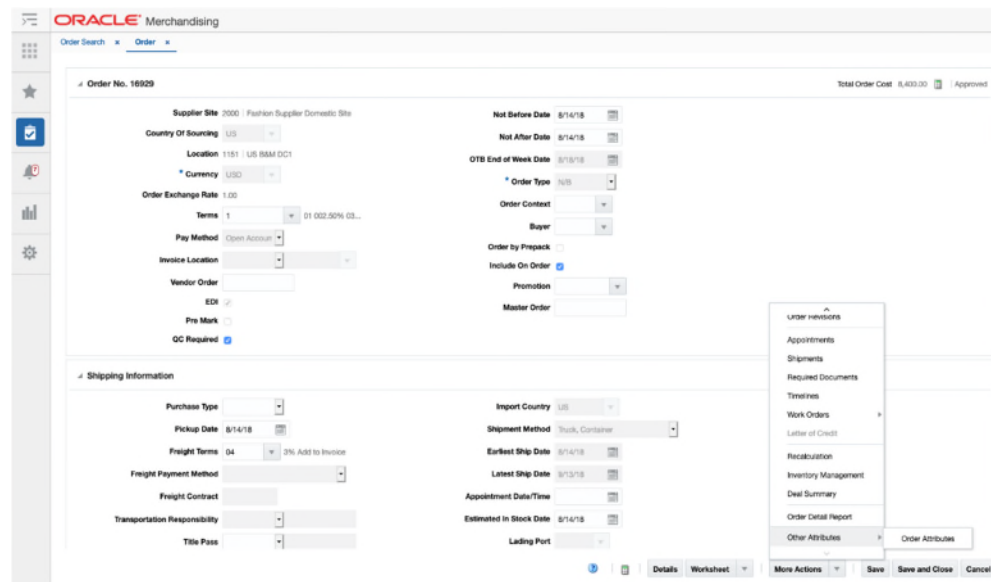
Custom Flex Attributes

Custom Flex Attribute Solution (CFAS) is a metadata driven framework that enables you to set up additional attributes on pre-enabled Merchandising entities without modifying base code. This functionality is available for both an on-premise and cloud service implementation of Merchandising, although there are certain features that are available only for on-premise implementations. This will be described in more detail later in the document.

The CFAS framework enables you to extend Merchandising entities to account for attributes that you require for your business, such as to support reporting or integration requirements. The entities that support flex attributes are:

Entity	Table Name
Address	ADDR
Class	CLASS
Cost Changes	COST_SUSP_SUP_HEAD
Cost Components	ELC_COMP
Deals	DEAL_HEAD
Department	DEPS
Diff Types	DIFF_TYPE
Item	ITEM_MASTER
Item Location	ITEM_LOC
Item Supplier	ITEM_SUPPLIER
Item Supplier Country	ITEM_SUPP_COUNTRY
Item Supplier Country Location	ITEM_SUPP_COUNTRY_LOC
Purchase Order	ORDHEAD ORDSKU ORDLOC
Partners	PARTNER
Return to Vendor	RTV_HEAD
Store	STORE
Suppliers	SUPS
Transfers	TSFHEAD
VAT Codes	VAT_CODES
Warehouses (physical and virtual)	WH

Once enabled for an entity, the flex attributes can be accessed using the More Actions menu in the relevant entity screen. The following figure illustrates how the option is displayed to users when attributes have been activated for purchase orders. The label displayed in the More Actions menu, in this case “Order Attributes”, is also part of the CFAS configuration. The attribute screens do not have special security associated with them. If a user can edit the main entity screen, they will also be able to edit the flex attributes associated with the entity.



In addition to managing the flex attributes in the Merchandising UI, these attributes can also be included on inbound integration if the source of the information is another application. And they are also published outbound so that they can be communicated to other dependent solutions.

Note: When an entity record is deleted, the related CFAS attributes associated with that entity will also be deleted.

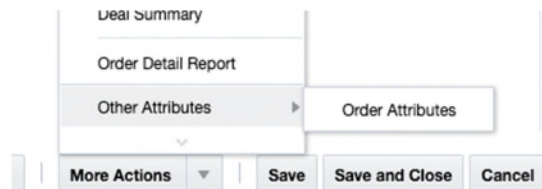
Defining Custom Flex Attributes

This chapter describes how flex attributes are organized so that you can plan for the best way to use the functionality to support your business and users. For example, it will be important to group together attributes that are being maintained by the same users. You should also consider structuring your attributes in such a way that you add more in the future as your business changes.

To help with the organization of attributes, a hierarchy is used, which consists of three levels: group set, group, and attribute. Group set is the name of the page, when the attribute screen is opened. Group level is shown as a container on the page, containing the attributes.

Group Sets

Group set is the highest level of the CFAS hierarchy. This is the level that is displayed in the Other Attributes menu where the attributes are accessed for each entity.



For each entity, you can define up to 99 attribute group sets. For each group set, you must define the following:

- **Display Order** - The order in which the attribute group set will appear in the More Actions menu of the relevant entity.
- **View Name** - The name that will be used for the database view that will be created for this group set. A view is a required information for each group set and is used for querying the data in Merchandising, as well as for integration from and to external solutions.
- **Staging Table Name** - The name that will be used for the staging table that will be created for this group set. Although staging table information is optional, it is recommended that you provide staging table name to support data loading from integrated systems. Each group set can have one staging table.
- **Labels** - Sets the business name for the group set. The label is the title that will appear in the Option menu for the entity when the attribute group set is accessed. You must set at least one label for the primary language for each group set you create. You can choose to create labels for more (alternate) languages, based on your business need.

Additionally, in an **on-premise implementation**, you can define validation functions for the group set. There are three types of validation that can be defined:

- Qualifier
- Default
- Validation

For details on how each of these functions are used, see [Custom Functions](#).

Groups

Once you determine the attribute group sets needed for each entity, the next step is to determine how the attributes will be organized within these sets. The attributes themselves are organized into groups, which is the middle layer of the hierarchy. Although you can create as many attribute groups as you want for each group set, you can only have 25 attributes in each attribute group.

When planning the attribute groups, in order to properly validate and display the information in the screens, you must determine the following in addition to the attributes themselves:

- **Display Order** - The order in which the attribute group set will appear in the Flex Attribute screen menu of the relevant entity.
- **View Name** - The name that will be used for the database view that will be created for this group. Similar to the view defined at the group set level, this view will contain all attributes in the group. A view is a required information for each group and is used to facilitate querying attribute information for the group.
- **Labels** - Sets the business name for the group and will be the name that appears on the screen. You must set at least one label for the primary language for each group set you create. You can choose to create labels for more (alternate) languages based on your business need.

Managing Entities, Group Sets and Groups

To add, update, or remove CFAS group sets or groups, or to update the entity details, select the template type of Administration from the Download Data screen and then the template Custom Flex Attribute Foundation. Click the Download button and when prompted, choose to either open the .ods file that is generated or save the file and open it separately in the spreadsheet application of your choice. There will be several tabs in the workbook that is generated that is used for managing the configuration of entities, group sets, and groups.

Add a Group Set

To add a group set, select the action type of Create in an empty row in the CFA Group Sets tab in the workbook. Next, enter a unique number of up to 10 digits as the Group Set ID. Then, select the RMS base table name where the group set will be associated, the order you want it displayed in the Merchandising UI, and define the name you want to use for the view and staging tables that will be created for the group set when activated. The view and staging table names can be up to 30 characters in length. Finally, you can optionally define custom qualifier, validation, and/or default functions, if you are implementing Merchandising on-premise. The custom function names must not exceed 61 characters and should be in the format of `package_name.function_name`.

For each group set created, you will also need to define at least one label in your primary language. To do this, navigate to the CFA Group Set Labels tab in the workbook. In a blank row in the worksheet, select an action type of Create and then enter the group set ID you used for your group set. Then, select your primary language configured in Merchandising and enter the label name you want used in the Label field. You can create labels of up to 255 characters, but it is recommended to try not to exceed 60 characters for best display in the Merchandising screens. Repeat this process for all languages you require for your users.

Add a Group

To add a group, select the action type of Create in an empty row in the CFA Groups tab in the workbook. Next, enter a unique number of up to 10 digits as the Group ID. Then, select the group set where the group will be associated, the order you want it displayed in the Merchandising UI, and define the name you want to use for the view that will be created for the group when activated. The view name can be up to 30 characters in length.

For each group created, you will also need to define at least one label in your primary language. To do this, navigate to the CFA Group Labels tab in the workbook. In a blank row in the worksheet, select an action type of Create and then enter the group ID you used for your group. Then, select your primary language configured in Merchandising and enter the label name you want used in the Label field. You can create labels of up to 255 characters, but it is recommended to try not to exceed 60 characters for best display in the Merchandising screens. Repeat this process for all languages you require for your users.

Modify an Entity, Group Set, or Group

If you would like to update any details for an entity, group set, group, or record group, a similar process will be followed as that described above for creating new. First, download the spreadsheet, and then navigate to the tab where you would like to make your updates. In the tab where you are going to make your updates, select the action type of Update, and then correct the value in the spreadsheet. The following columns can be updated in each tab:

- CFA Entity – add a custom function in the Validation Function column using the full package function name (package_name.function_name)
- CFA Group Sets – Display Order, Qualifier Function, Validation Function, and Default Function; also, Group Set View Name and Staging Table Name prior to attributes being activated for the group set.
- CFA Group Set Labels – Label
- CFA Groups – Display Order; also, Group View Name prior to attributes being activated for the group.
- CFA Group Labels - Label

Note: Once attributes are active, the data that can be updated for each of these areas is limited to the following: labels, display order, and custom functions. For custom functions, the changes will only be applicable going forward for editing or creating data in the impacted entity. It will not revalidate all the data that was previously created.

Delete a Group Set

If you wish to remove a group set, navigate to the CFA Group Set tab in the spreadsheet and select the Delete action on the row of the group set, you wish to delete. You must also remove all labels that have been associated with that group set in the CFA Group Set Labels tab, by selecting the Delete action in the row associated with those labels. Group sets cannot be deleted if the attributes in the group set have been activated, however labels can be removed at any time.

Group sets cannot be deleted if they are active.

Delete a Group

If you wish to remove a group, navigate to the CFA Groups tab in the spreadsheet and select the Delete action on the row of the group, you wish to delete. You must also remove all labels that have been associated with that group in the CFA Group Labels tab, by selecting the Delete action in the row associated with those labels. Groups cannot be deleted if the attributes in the group have been activated, however labels can be removed at any time.

Groups cannot be deleted if they are active.

Uploading Changes

For all actions defined above, once all the updates have been made to the data in the spreadsheet, save the file and close it. Then, return to the Merchandising screens and select Foundation Data > Upload Foundation Data from the main task list. In this screen, you'll again select the template type Administration and the template Custom Flex Attribute Foundation. This will generate a process description automatically, but this can be updated if desired. Lastly, select the Browse button and navigate to the directory where you saved the updated spreadsheet.

To review the status of the upload and check whether any errors occurred, select the Foundation Data > Review Status task from the main task list.

See also the *Oracle Retail Merchandising Do the Basics User Guide* section on Download/Upload Data from Spreadsheets for more information.

Attributes

Attributes are the bottom layer of the hierarchy. As mentioned above, you can have 25 attributes per group. Of those 25 attributes, up to 10 can be character-based attributes, up to 10 can be number-based attributes, and up to 5 can be date attributes. You should consider this limit when planning the attributes to be included in each group.

The screenshot shows the Oracle Merchandising application interface. The main content area displays the configuration for Custom Flex Attributes. At the top, there are tabs for 'Order Search', 'Order', 'Custom Flex Attributes', and 'Order Attributes'. Below the tabs, there are dropdown menus for 'Application Table' (set to ORDHEAD) and 'Custom Extension Table' (set to ORDHEAD_CFA_EXT). A 'Group Set' dropdown is set to 'Order Attributes', and a 'Group Set Active' checkbox is checked. There are buttons for 'Display Attributes' and 'Activate'.

The 'Attributes' section contains a table with the following data:

Group	View Column Name	Display Order	Data Type	Widget Type	Maximum Length	Required	Enabled	Active
Order Attributes	SEASON	1	NUMBER	List of Values	5	—	✓	✓
Order Attributes	COLLECTION	2	NUMBER	List of Values	5	—	✓	✓
Order Attributes	ON_FLOOR_DATE	3	DATE	Date	—	—	✓	✓
Order Attributes	TICKETS_SENT	4	VARCHAR2	Check Box	1	—	✓	✓
Order Attributes	TICKETING_PARTNER	5	NUMBER	List of Values	5	—	✓	✓

Below the attributes table, there is a 'Labels' section with a table for Label and Language:

Label	Language
Season	English

At the bottom right of the screen, there are buttons for 'View UI', 'Save and Close', and 'Cancel'.

To start specifying attributes, enter the entity you want to add the attributes to in the Application Table field, and then select the group set you previously created, then click **Display Attributes**. Next, select the Add option in the Actions menu or select the iconic button to begin adding attributes.

When creating attributes, you'll need to specify the following:

View Column Name - this value will be used to define the name of the attribute in the group set and group level views, as well as the group set level staging table.

Label - you will be required to add at least one label for the attribute in the primary language. But, labels for other languages can also be added.

Display Order - Indicates the order in which the attributes will appear in the attribute screen, from top to bottom. Attributes will appear in a single column on the screen.

Maximum Length - Indicates the maximum number of digits or characters that are allowed in the attribute, as well as the width of the attribute on the screen. You must specify this information for the attributes with Number or Varchar data types. This may not apply when you choose a List Item or Check box as the widget type. The maximum length for the List Item widget type is automatically set to 6 and the maximum length for the Check box widget type is automatically set to 1. For attributes with Number data type, you must enter the full length of the number. This includes the length to be allowed after the decimal point and positive/negative sign (if negative numbers are allowed). For example, if a particular attribute needs to allow for five digits with up to two digits after the decimal point and allow negative integers, you will need to specify the length as 9 (1 character for positive/negative sign, 5 digits before the decimal point, 1 decimal point, and 2 digits after the decimal point).

Note: If you choose a record group, the maximum length will be automatically selected based on the value in the first column of the record group query.

Minimum/Maximum Value Allowed - Optional and applies to attributes with Number or Date data types. Indicates the minimum and maximum numeric or date value that can be entered for the attribute. For date widgets, the minimum and maximum definition is the number of days.

For example, if an attribute was added for an item that was Ecommerce Launch Date and the attribute was set like this:

- Lowest Allowed Val: 0
- Highest Allowed Val: 10

If the current VDATE is 12-SEP, then the min/max setup would not allow this attribute to be earlier than 12-SEP or greater than 22-SEP.

Data Type - Indicates the type of data for the attribute. You can set this as a Number, Varchar, or Date.

Widget Type - Indicates the type of the field that will appear for the attribute. You can select one of the following options:

- Text Item - used for both Number and Varchar data types. When used, the attributes field will appear as a text box on the screen.
- List of Values (LOV) - used for the Varchar data types only. A list of values appears as combo box LOV. If you choose to use this widget type, you must also specify a [record group](#).
- List Item - used for the Varchar data type only. When used, the attributes field will appear as a select one choice dropdown on the screen. If you choose to use this widget type, you must also specify a code type in the List Item Code Type field. The valid codes types are those that exist on the Codes and Descriptions table in Merchandising. For more information on Codes and Descriptions, see the *Oracle Retail Merchandising Implementation Guide*.
- Check box - used for Varchar data type only. When used, the attributes field will appear as a check box on the screen.
- Date - used for Date data type only. When used, the attributes field will appear with the calendar icon to allow users to select dates.

Required - Indicates whether the attribute will be considered as a mandatory field. Once an attribute is activated, it is recommended that you avoid changing this information.

Wrap Text - used for data type of VARCHAR and widget type Text Item, this may be used if you expect to have longer than usual text entered for your text item and want the widget to support wrapping text on the screen.

Enabled - Indicates whether the attribute appear as enabled or disabled (greyed out); only enabled attributes can be updated in the attribute screen. For attributes where you want the users to enter information, you must set them as enabled. For attributes that will display a default value (using a default function at the group set level), you will need to set them as disabled.

Additionally, in an on-premise implementation, you can define a validation function for the attribute, if you wish to validate specific conditions for the attribute when added. In such a case, you will need to write a custom package function for the validation function based on your business need.

Once you have specified the attributes to be added, you can choose the View UI button to see an example of what the attributes will look like once they are activated. If needed, make any changes to display order, labels, and so on. When you are ready to activate the attributes, which will make them visible to users, click the **Activate** button. This also generate the views at the group set and group level as you have defined them.

Custom Functions

There are three types of custom functions that CFAS is designed to support, however these are only able to be used for **on-premise Merchandising implementations** because, in order to use this capability, you will need to write a custom package function in order to perform the functions described below. All are optional components of using the CFAS functionality.

Qualifier

Qualifier functions are at the group set level only and are used to ensure that the required information for the calling entity is entered or available before the users open the flex attribute screen. It is executed when the user selects the menu option associated with the group set from the More Actions list from the entity. The base solution's validation ensures that the basic information for the entity is entered before launching the attribute screen, so this function would be needed only if you require something over and above that validation in order for your users to add or modify the attributes.

For example, in the Ordering dialog, you may use a qualifier function to ensure that at least one item is added to the order before accessing one of the CFAS screens from the Order Header screen.

Default

Default functions are also only at the attribute group set level. These types of functions can be used default values for the attributes in the group set based on some previously determined criteria. For example, copying information from the subclass level to the item level when the same attributes are set up for both the entities.

Once implemented, the default function will run each time the users open the relevant screen. Therefore, you must ensure that the custom function is written in such a manner that it only sets the default values when attributes are set up for the first time in an entity and will not cause a user defined value to be overwritten.

Validation

Validation functions can be defined at the entity, group set, and attribute level. These functions enable you to add additional validation into the attribute screens above the default validation, which is based on how the attributes are created (required, minimum or maximum values, and so on).

Since the validation functions work slightly different based on the level at which it is set, determining the level for the validation is an important part of the planning process. Validation for the entity is triggered when changes are saved in the entity, usually when clicking Save or Save and Close. Validation at the group set level is executed when users click the Save and Close on the attribute screen. Validation at the attribute level is executed when users navigate to other attribute fields by pressing the Tab key.

Syntax

If there is a complex validation defined for the attribute in the metadata, it executes the function dynamically. The validation functions must be defined to have only one parameter – O_ERROR_MESSAGE.

The body of a validation function can be subdivided into the following three areas:

- **Input** - This section acts as the input parameter for the function. The section may have multiple calls to CFA_SQL.GET_VALUE to get the parameters needed for the validation. Values using this function are limited to the header field values (or the

CFA_EXT_ENTITY_KEY.KEY_COL) and the attribute fields values in any group within the group set. If the function requires other values (other than these such as system options or values from the main RMS base table, and so on), the function needs to define a cursor to fetch these information.

- **Validation Logic** - This section is where the actual complex validation takes place. The validation logic can be written out directly or make calls to regular Merchandising package functions.
- **Output** - If the validation logic function returns a value that needs to be used to update a field, such as description or other fields that are defaulted, calls to the CFA_SQL.SET_OUTPUT function are done in this section.

Example Function

```
FUNCTION GET_ITEM_DESC (O_error_message IN OUT RTK_ERRORS.RTK_TEXT%TYPE)
RETURN BOOLEAN AS
```

```

L_program    VARCHAR2(62) := 'CUSTOM_CFA_DEMO_SQL.GET_ITEM_DESC';
L_field      CFA_SQL.GP_FIELD_ITEM%TYPE := 'ITEM';
L_item       ITEM_MASTER.ITEM%TYPE;
L_desc       ITEM_MASTER.ITEM_DESC%TYPE;
L_data_type  CFA_SQL.GP_DATA_TYPE%TYPE;
```

```
BEGIN
```

```
-----
-- Get keys and/or attribute value inputs in this section.
-----
```

```
if NOT CFA_SQL.GET_VALUE(O_error_message,
                        L_item,
                        L_desc, -- initially NULL
                        L_data_type,
                        L_field) then
    return FALSE;
end if;
```

```
-----
-- Validation logic in this section. It can be a call to a non-CFAS package
-- function.
-----
```

```
if L_desc is NULL then
    if NOT ITEM_ATTRIB_SQL.GET_DESC(O_error_message,
                                    L_desc,
                                    L_item) then
        return FALSE;
    end if;
end if;
```

```
-----
-- If there are outputs set each in this section
-----
```

```
if NOT CFA_SQL.SET_OUTPUT(O_error_message,
                          CFA_SQL.CFA_KEY,
                          L_field,
                          L_item,
                          L_desc) then
    return FALSE;
```

```
end if;
return TRUE;
EXCEPTION
when OTHERS then
```

```

O_error_message := SQL_LIB.CREATE_MSG('PACKAGE_ERROR',
                                       SQLERRM,
                                       L_program,
                                       to_char(SQLCODE));

return FALSE;
END GET_ITEM_DESC;

```

The sample function above gets the description for the item. It uses the CFA_SQL.GET_VALUE to get the item value from the content collection. If more parameters are required, several calls to this function are required for each parameter.

The validation logic section can be any business validation written directly in the section or called from another package.

If the validation needs to output something like a description or update another attribute field in any group within the group set, a call to CFA_SQL.SET_OUTPUT can be made. In the above example, the item description is returned by the RMS function ITEM_ATTRIB_SQL.GET_DESC. The CFA_SQL.SET_OUTPUT takes the description value and updates the content collection.

Managing Record Groups

To add, update, or remove record groups, select the template type of Administration from the Download Data screen and then the template Custom Flex Attribute Foundation. Click the Download button and when prompted, choose to either open the .ods file that is generated or save the file and open it separately in the spreadsheet application of your choice. There will be two tabs in the workbook that is generated that is used for managing the configuration of record groups.

Add a Record Group

If you plan to add any attributes that are of type List of Values, then you will need to define a record group as part of the configuration. To add a record group, select the action type of Create in an empty row in the CFA Record Groups tab in the workbook. Next, enter a unique number of up to 10 digits in the Record Group column and enter a name in the Record Group Name column of up to 30 characters in your configured primary language. Next, you'll choose the query type of either simple or complex. A simple query will be generated systematically based on the addition of attributes in the spreadsheet. Whereas a complex query can include more conditions and must be defined outside this worksheet.

Note that in a **cloud service implementation**, only the simple query type is supported. For simple queries, you'll need to include the following information:

- Table Name – the table against which the record group query will execute; this must be a valid table name in the Merchandising schema - for example, UDA_VALUES. This can be up to 30 characters in length.
- Value Column – this will be part of the generated select statement in the record group that will return an entity ID; it is usually the primary key of the table specified in the Table Name column (such as, UDA_VALUE). This can be up to 30 characters in length.
- Description Column – this will be part of the generated select statement in the record group that will return a description that corresponds to the value column; it is usually the description of the entity as defined in the table name (such as, UDA_VALUE_DESC). This can be up to 30 characters in length.

- Column 1 - the three values here will be used to generate a where clause if you want the record group to return only a subset of the table. This is optional for all simple queries, but if one value is defined for column 1, then all three must be defined.
 - Where - column name that should be used to limit the values returned (e.g. UDA_ID). This can be up to 30 characters in length.
 - Operator - valid values are !=, <, <=, =, >, >=, is NULL, or is not NULL
 - Condition - indicates the condition that will limit the results. For the UDA example, this may be the specific UDA ID that should be returned. This can be up to 120 characters in length.
- Column 2 - this is optional for all simple queries, but if one value is defined for column 2, then all three of where, operator, and condition must be defined.

For complex record groups, you will need to insert the query you want direction into the CFA_RECORD_GROUP table in Merchandising. So, the columns listed above are not required.

For each record group created, you will also need to define at least one set of labels in your primary language. To do this, navigate to the CFA Record Group Labels tab in the workbook. In a blank row in the worksheet, select an action type of Create and then enter the record group ID you used for your record group.

Then, select your primary language configured in Merchandising and enter the label names you want used in the LOV Title, Value Column, and Desc Column Header fields. You can create labels of up to 255 characters, but it is recommended to try not to exceed 60 characters for best display in the Merchandising screens. Repeat this process for all languages you require for your users.

Modify a Record Group

If you would like to update any details for an entity, group set, group, or record group, a similar process will be followed as that described above for creating new. First, download the spreadsheet, and then navigate to the tab where you would like to make your updates. In the tab where you are going to make your updates, select the action type of Update, and then correct the value in the spreadsheet. The following columns can be updated in each tab:

- CFA Entity - add a custom function in the Validation Function column using the full package function name (package_name.function_name)
- CFA Group Sets - Display Order, Qualifier Function, Validation Function, and Default Function; also, Group Set View Name and Staging Table Name prior to attributes being activated for the group set.
- CFA Group Set Labels - Label
- CFA Groups - Display Order; also, Group View Name prior to attributes being activated for the group.
- CFA Group Labels - Label
- CFA Record Groups - no changes can be made once the attribute using this record group is active; prior to activation, the following columns are editable: Record Group Name, Query Type, Table Name, Value Column, Description Column, Where Column 1, Operator 1, Condition 1, Where Column 2, Operator 2, Condition 2
- CFA Record Group Labels - LOV Title, Value Column, Desc Column Header

Note: Once attributes are active, the data that can be updated for each of these areas is limited to the following: labels, display order, and custom functions. For custom functions, the changes will be applicable going forward for editing or creating data in the impacted entity. It will not revalidate all the data that was previously created.

Delete a Record Group

If you wish to remove a record group, navigate to the CFA Record Groups tab in the spreadsheet and select the Delete action on the row of the record group, you wish to delete. You must also remove all labels that have been associated with that record group in the CFA Record Group Labels tab, by selecting the Delete action in the row associated with those labels. Record groups cannot be deleted if the attribute using the record group has been activated, however labels can be removed at any time.

Uploading Changes

For all actions defined above, once all the updates have been made to the data in the spreadsheet, save the file and close it. Then, return to the Merchandising screens and select Foundation Data > Upload Foundation Data from the main task list. In this screen, you'll again select the template type Administration and the template Custom Flex Attribute Foundation. This will generate a process description automatically, but this can be updated if desired. Lastly, select the Browse button and navigate to the directory where you saved the updated spreadsheet.

To review the status of the upload and check whether any errors occurred, select the Foundation Data > Review Status task from the main task list.

See also the *Oracle Retail Merchandising Do the Basics User Guide* section on Download/Upload Data from Spreadsheets for more information.

Using Custom Flex Attributes

Flex attributes are not used in any base processing, but they can be included in custom reports, used in [custom approval rules](#), or integrated to other solutions. When querying flex attributes, it is recommended that you use the views at the group set or group level, so that the queries can be built more like for other tables. The views are automatically generated when the attributes are activated.

For integration, the flex attributes will also be automatically made available in inbound and outbound integration when they are activated. This is done slightly differently based on the method of integration.

For message-based integration via the Oracle Retail Integration Bus (RIB) and bulk data integration (BDI), attributes are published as name-value pairs based on the column name defined at the attribute level. This is true for outbound and inbound.

For spreadsheet upload and bulk loads that use those templates, the CFAS extension tables can also be enabled for addition to the templates. For entities like item and PO, where there is the ability to create your own templates, you will have the option to manually add the CFAS extension tables to your customized templates using the view name defined at the group set level. For other entities, like diff types, where customization of templates is not supported, the flex attributes are added automatically to the template when activated. For more information on customizing spreadsheet upload templates, see the Configure Spreadsheet Download/Upload section in the *Oracle Retail Merchandising Implementation Guide*.

Note: Deals Upload does not support uploading flex attributes. If you add attributes at that level, they can only be managed in the Merchandising UI.

In Context Launch

Merchandising solutions expose select task flows that you can directly access from external solutions using a specific URL, custom reports, and so on. This feature is referred to as in-context launch because, as part of the URL, parameters for the workflow are passed in to open a page displaying information related to those parameters. For example, the Item task flow in Merchandising could be called for a specific item ID. The requested task flow will be displayed in a new browser window or tab, depending on the specified target of the URL.

Merchandising

Item Header

When this page is launched, Merchandising will display the information for the item included in the URL in either view or edit mode.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmItem=<item>&pmMode=<mode>&navModelItemId=MaintainItemFlowExt`

Parameters:

Parameter	Required	Description
Item	Yes	Item ID
Mode	No	The mode in which the screen should be opened. Valid values are EDIT or VIEW.

Order Search

When this page is launched, Merchandising will display the search results based on the hierarchy values sent in the URL. One or more departments can be sent, or a single department/class or single department/class/subclass combination. If none of the parameters are sent, the screen will be displayed with no search results.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmDepts=<dept>&pmClass=<class>&pmSubClass=<subclass>&navModelItemId=searchOrderflowExternal`

Parameters:

Parameter	Required	Description
Department	No	One or more department IDs; if multiple sent, they should be separated by commas. For example, pmDepts=100,200,300.
Class	No	Class ID
Subclass	No	Subclass ID

Order

When this page is launched, Merchandising will display information for the order included in the URL in either view or edit mode.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmMode=<mode>&pmOrderNo=<order>&navModelItemId=MaintainPurchaseOrderFlow`

Parameters:

Parameter	Required	Description
Order	Yes	Purchase order number
Mode	No	The mode in which the screen should be opened. Valid values are EDIT or VIEW.

Transfer

When this page is launched, Merchandising will display information for the transfer included in the URL in either view or edit mode.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmMode=<mode>&pmTsfNo=<transfer>&navModelItemId=MaintainTransferHeaderFlow`

Parameters:

Parameter	Required	Description
Transfer	Yes	Transfer number
Mode	Yes	The mode in which the screen should be opened. Valid values are EDIT or VIEW.

Shipment

When this page is launched, Merchandising will display information for the shipment included in the URL in either view or edit mode.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmShipment=<shipment>&pmMode=<mode>&navModelItemId=MaintainShipmentDetailFlow`

Parameters:

Parameter	Required	Description
Shipment	Yes	Shipment number
Mode	No	The mode in which the screen should be opened. Valid values are EDIT or VIEW.

Return to Vendor

When this page is launched, Merchandising will display the information for the RTV included in the URL in either view or edit mode.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmRtvOrderNo=<rtv>&pmMode=<mode>&navModelItemId=MaintainRtvDetailFlow`

Parameters:

Parameter	Required	Description
RTV	Yes	RTV number
Mode	No	The mode in which the screen should be opened. Valid values are EDIT or VIEW.

Transaction Data

When this page is launched, Merchandising will display the transaction data based on the search criteria included in the URL. In order to successfully execute the search, at least one of the following must be included department, item, store, or warehouse, along with the transaction date range.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmItem=<item>&pmWarehouse=<warehouse>&pmStr=<store>&pmTransStrtDate=<tran_start_date>&pmTransEndDate=<tran_end_date>&pmPostStrtDate=<post_start_date>&pmPostEdDate=<post_end_date>&pmCreateDate=<create_date>&pmDpt=<dept>&pmCls=<class>&pmSubcls=<subclass>&navModelItemId=ViewTransactionDataFlow`

Parameters:

Parameter	Required	Description
Item	No	Item ID
Warehouse	No	Virtual warehouse ID
Store	No	Store ID
Transaction Start Date	Yes	Start range for searching on transaction date
Transaction End Date	No	End range for searching on transaction date; if not included in the URL, the start date will be used as the end date in the search.
Post Start Date	No	Start range for searching on post date
Post End Date	No	End range for searching on post date
Create Date	No	Create Date
Department	No	Department ID
Class	No	Class ID
Subclass	No	Subclass ID

Average Cost Adjustments

When this page is launched, Merchandising will display the average cost adjustment screen with the item and location data defaulted based on the information in the URL.

URL:

`http://<host>:<port>/Rms/faces/RmsHome?pmMode=<mode>&pmItem=<item>&pmLocType=<loc_type>&pmLocation=<location>&navModelItemId=MaintainAvgCostAdjFlowExt`

Parameters

Parameter	Required	Description
Item	Yes	Item ID
Location Type	Yes	Location type - either S (store) or W (warehouse)
Location	Yes	Store or virtual warehouse ID
Mode	No	The mode in which the screen should be opened. Valid values are EDIT or VIEW.

Allocation**Quick Create**

When this page is launched, Allocation will open the Allocation Maintenance page to initiate an allocation with the passed in source type, source ID and item IDs.

URL:

```
https://<host>:<port>
/alloc/faces/Home?navModelItemId=quickCreateTF&source=<source>&sourceIdList=<source_list>&itemIdList=<item_list>
```

Parameters:

Parameter	Required	Description
Source	Yes	Should be a valid source type for an allocation - PO, WAREHOUSE, ASN, or BOL
Source ID List	Yes	Should indicate the entity ID that relates to the source. For example, if the source is PO, then this would be one or more purchase order number. If more than one is included, they should be separated by semicolons.
Item List	Yes	Should indicate the items that will be allocated. If more than one is included, they should be separated by semicolons.

Load Allocation

When this page is launched, Allocation will open the allocation included in the URL in edit mode.

URL:

```
http://<host>:<port>
/alloc/faces/Home?navModelItemId=loadAllocationTF&allocationId=<alloc_id>
```

Parameter

Parameter	Required	Description
Allocation	Yes	Valid allocation ID.

Invoice Matching

Summary Match

When this page is launched, Invoice Matching fetches all the invoices and receipts associated with the default match key from the supplier and for the given invoice.

URL:

`http://<host>:<port>/ReimViewController/faces/Home?navModelItemId=SummaryMatchService&supplier=<supplier ID>&invoice=<invoice ID>`

Parameter

Parameter	Required	Description
Supplier	Yes	Supplier ID
Invoice	Yes	Invoice or document ID

Discrepancy List Flow

When this screen is launched, Invoice Matching fetches all the discrepant invoices for the given supplier and invoice.

URL:

`http://<host>:<port>/ReimViewController/faces/Home?navModelItemId=DiscrepancyListService&supplier=<supplier ID>&invoice=<invoice ID>`

Parameters:

Parameter	Required	Description
Supplier	Yes	Supplier ID
Invoice	Yes	Invoice or document ID

Sales Audit

Store Day Search

When this page is launched, Sales Audit will pre-populate the Store Day search based on the parameters included and optionally auto execute the search. Only one store/day combination can be included.

URL:

`http://<host>:<port>/ResaPortal/faces/Home?navModelItemId=SearchStoreDayTF?Store=<storeId>&BusinessDay=<BusinessDay>`

Parameters:

Parameter	Required	Description
Store	Yes	The ID for a store to include in a search
BusinessDay	Yes	Business day to use in the search in the format DD-MON-RR.
AutoExecute	No	Indicates if the query should be automatically executed when the screen is launched or not. Valid values are Y or N.

Parameter	Required	Description
AssignedStores	No	Filters the returned list to the user's assigned stores (Y) or to the stores not assigned to the user (N).
DataStatus	No	Filters the returned list of store days by data status for one of the following: Fully Loaded (F), Purged (G), Loading (L), Partially Loaded (P), and Ready for Import (R).
OverAllStatus	No	Filters the returned list of store days by overall status for one of the following: Complete (A) or In Progress (I).

Store Day Maintenance

When this page is launched, Sales Audit will display information for the store day based on the parameters included in the URL in edit mode or view mode, depending on the user's privileges.

URL:

`http://<host>:<port>/ResaPortal/faces/Home?navModelItemId=MaintainStoreDayTF?StoreSeqNo=<StoreDaySequenceNo>&Store=<Store ID>&BusinessDate=<BusinessDate>`

Parameters:

Parameter	Required	Description
StoreSeqNo	Yes	The sequence number for the store day.
Store	Yes	ID for a store
BusinessDate	Yes	Business day to use in the search in the format DD-MON-RR.
TabToDisclose	No	This parameter, if included can indicate that the Over/Short tab should be the focus when the page opens. If this parameter is included, it must have a value of OS, which means that the Over/Short tab will be in focus. Otherwise, the focus will be on the error list.

Transaction Maintenance

When this page is launched, Sales Audit will display information for the transaction sequence number included in the URL in edit mode or view mode, depending on the user's privileges.

URL:

`http://<host>:<port>/ResaPortal/faces/Home?navModelItemId=MaintainTransactionTF?TransactionSeqNo=<TransactionSeqNo>`

Parameters:

Parameter	Required	Description
TransactionSeqNo	Yes	The sequence number for the transaction.

Transaction Search

When this page is launched, Sales Audit will pre-populate the Transaction search based on the parameters included and optionally auto execute the search. Only a single transaction sequence can be sent.

URL:

`http://<host>:<port>/ResaPortal/faces/Home?navModelItemId=ManageTransactionTF?TransactionSeq=<TransactionSeq>&Store=<StoreID>&TranBusinessDate=<TranBusinessDate>`

Parameters:

Parameter	Required	Description
TransactionSeq	Yes	The sequence number for the transaction.
StoreId	Yes	The ID of a store to include in the search.
TranBusinessDate	Yes	The business date to use in the search in the format in the format DD-MON-RR.
ErrorExists	No	If included, filters the returned list to transactions with errors (Y) or without (N).
AutoExecute	No	Indicates if the query should be automatically executed when the screen is launched or not. Valid values are Y or N.

Custom Rules

Custom Merchandising Rules

Because different retailers have different rules that they want to apply for their business in order to meet their own specific business processes, Merchandising supports the ability for retailers to configure custom rules for certain areas, for **on premise implementations**, by providing an API that can be used to call a PL/SQL function. This process allows for configuration without modification to base code. Currently, this is supported for submit and approval of certain functions like items and purchase orders, as well as for VAT calculations when Merchandising is configured to run in a Simple VAT configuration. Some examples of how this could be used include:

- Ensuring all transaction level items are not approved before reference item has been created for the item.
- Preventing purchase orders from being approved if a factory has not yet been associated with the order.
- Using alternative rates for calculating tax for certain items based on custom flex attributes (CFAS), when running RMS in a simple VAT configuration.

Approval Rules

The following functional areas support the definition of custom rules:

- Item Submit and/or Approve
- Purchase Order Submit and/or Approve
- Transfer Submit and/or Approve
- Mass-return Transfer Submit and/or Approve
- Return to Vendor Approve
- Inventory Adjustment Creation
- Cost Change Submit and/or Approve
- Transportation Finalization
- Replenishment Attribute Activation and/or Deactivation
- Supplier Activation and/or Deactivation
- Partner Activation and/or Deactivation

It would need to supply the package function that contains the additional validations that should be performed for any areas you want to add custom rules. The function needs to follow a specific format:

Parameter	Input/Output	Type	Comments
O_ERROR_MESSAGE	In/Out	RTK_ERRORS.RTK_KEY	Errors returned from the function must exist on the RTK_ERRORS table in Merchandising to ensure that messages are properly displayed in standard error handling.
O_ORD_APPRERR_EXIST	In/Out	BOOLEAN	This is only used for PO approval package functions. It works with the error message parameter and in the case of any custom validation failures, the parameter should be set to TRUE and the error message parameter should be populated accordingly.
IO_CUSTOM_OBJ_REC	In/Out	CUSTOM_OBJ_REC	This is PL/SQL TYPE object which has placeholders to pass the input parameters to the custom code. The calling function will pass the following to this object: Function key Call sequence number Entity ID (such as, item number)

Additionally, the function needs to be structured to comply with the following:

- The custom function return type must be BOOLEAN. No other data type is allowed.
- In case of fatal errors, the custom function should return FALSE along with an appropriate error message in the output variable. Any non-fatal errors should return TRUE with the appropriate error message included if the validation rules are violated.
- Errors for item approval should be coded to write to the item approval errors (ITEM_APPROVAL_ERROR) table, similar to base approval rule violations. This will ensure that the approval rules are applied seamlessly for the end user.
- When using the Order Subscription API or the Store Order Subscription API, the errors will be propagated to the calling function. Approval errors encountered during other methods of PO induction (such as, spreadsheet upload, bulk upload) are logged in an error table, together with the other order creation validation issues.

Here is an example of a package specification for items:

```
CREATE OR REPLACE PACKAGE MY_OWN_ITEM_APPROVAL AS
-----
This function checks whether items that are to be approved have comments
FUNCTION CHECK_COMMENTS (O_error_message IN OUT RTK_ERRORS.RTK_TEXT%TYPE, IO_custom_obj_rec
IN OUT CUSTOM_OBJ_REC)

RETURN BOOLEAN;
-----
END MY_OWN_ITEM_APPROVAL;
```

Here is an example of a package specification for POs:

```
CREATE OR REPLACE PACKAGE MY_OWN_APPROVAL_RULES AS
-----
This function checks whether orders exist.
FUNCTION CHECK_PRESENCE (O_error_message IN OUT RTK_ERRORS.RTK_TEXT%TYPE, O_apprerr_exist
IN OUT BOOLEAN,
IO_custom_obj_rec IN OUT CUSTOM_OBJ_REC)
RETURN BOOLEAN;
-----
END MY_OWN_APPROVAL_RULES;
```

Configure your custom package functions to be called by the base logic for each functional area by adding records to the CUSTOM_PKG_CONFIG table in Merchandising. The key components of this configuration are the following columns in the table:

Column Name	Description
FUNCTION_KEY	See below
CALL_SEQ_NO	This indicates the sequence of the custom code to be executed. This must be set to 1. If you have more than one function that needs to be called it is recommended that these be called in the correct sequence from the package function indicated in this table.
SCHEMA_NAME	The schema where the custom package function is compiled.
PACKAGE_NAME	This should be the name of your custom package, if applicable. This should be left blank if the custom code is a function without a package.
FUNCTION_NAME	The name of your custom function or package function within the package noted above. Only PL/SQL functions are allowed as custom code in this table, either standalone PL/SQL functions or as part of a package.

Function Keys

The function key column in the configuration table determines where the function will be called from. In some cases, the function can be called differently based on where the approval is being initiated. Here are how the keys should be set for each of the supported functional areas:

Function Key	Description
COST_CHANGE_APPROVE_UI	This will be executed when approving a cost change in the UI, or any method of cost change approval

Function Key	Description
COST_CHANGE_SUBMIT_UI	This will be executed when submitting a cost change in the UI, or any method of cost change submission.
INV_ADJ_UI	This will be executed when performing an inventory adjustment in the UI, or any method of adjusting inventory.
ITEM_APPROVE_UI	This will be executed for item approval whether items are approved in the UI or any method of item induction (spreadsheet, RIB, web service, bulk).
ITEM_SUBMIT_UI	This will be executed for item submission whether items are submitted in the UI or any method of item induction (spreadsheet, RIB, web service, bulk).
MRT_SUBMIT_UI	This will be executed when submitting an MRT in the UI, or any method of submitting an MRT.
MRT_APPROVE_UI	This will be executed when approving an MRT in the UI, or any method of approving an MRT.
ORDER_APPROVE_RPL	This will be executed by replenishment processes when approving orders.
ORDER_APPROVE_UI	This will be executed when approving orders in the UI.
ORDER_APPROVE_POI	This will be executed when orders are approved as part of the Store Orders API or in any method of PO Induction (RIB, bulk, web service, spreadsheet)
REPLENISHMENT_ACTIVATE_UI	This will be executed when activating a replenishment attribute in the UI, or any method of activating a replenishment.
REPLENISHMENT_DEACTIVATE_UI	This will be executed when deactivating a replenishment attribute in the UI, or any method of deactivating a replenishment.
TRANSPORTATION_FINALIZE_UI	This will be executed when finalizing a transportation record in the UI, or any method of finalizing a transportation record.
RTV_APPROVE_UI	This will be executed when approving an RTV in the RMS screen, or any method of approving an RTV
PARTNER_ACTIVATE_UI	This will be executed when activating a partner in the UI, or any method of activating a partner.
PARTNER_DEACTIVATE_UI	This will be executed when deactivating a partner in the UI, or any method of deactivating a partner.
SUPPLIER_ACTIVATE_UI	This will be executed when activating a supplier in the UI, or any method of activating a supplier.

Function Key	Description
SUPPLIER_DEACTIVATE_UI	This will be executed when deactivating a supplier in the UI, or any method of deactivating a supplier.
TSF_APPROVE_UI	This will be executed when approving transfers in the UI, or any method of approving a transfer.
TSF_SUBMIT_UI	This will be executed when submitting transfers in the UI, or any method of submitting a transfer.

Custom VAT

For Simple VAT implementations, you can configure custom tax processing for any region that has a tax calculation type defined as Custom. The custom logic would be executed in the following scenarios based on the VAT regions of the locations involved:

Location A Tax Calculation	Location B Tax Calculation	Tax Calculation
Simple	Simple	If Entities A and B lie in the same VAT Region, base simple VAT rules used. If different regions, VAT calculated as zero.
Simple	Exempt	VAT calculated as zero
Exempt	Simple	VAT calculated as zero
Exempt	Exempt	No VAT calculation performed
Simple	Custom	Custom function executed
Exempt	Custom	Custom function executed
Custom	Custom	Custom function executed
Custom	Simple	Custom function executed
Custom	Exempt	Custom function executed
Custom	Custom	Custom function executed

* Location could refer to a store, warehouse, partner, supplier, or other entity.

Note: For VAT regions flagged as custom, VAT codes and rates are expected to be set up, even though the VAT code may not solely determine the tax behavior in all cases.

Conceptually, custom VAT rules are configured similar to the rules described above, but a different function is used for the custom rules. For VAT, the custom rules are called by the CUSTOM_TAX_SQL package. This package contains the below functions:

- CALC_CTAX_COST_TAX - which is used for cost-based VAT calculations
- CALC_CTAX_RETAIL_TAX - which is used for retail-based VAT calculations
- ADD_CTAX_RETAIL_TAX - which is used for mark-up calculations

Inputs to the function are:

- Item
- From Location
- To Location

- Tax Type - Cost, Retail, or Both
- Transaction ID (such as, PO number)

The outputs from the function call are:

- Tax Code
- Tax Rate
- Tax Amount
- Tax Exempt flag (Y or N)
- Error Message

The function body for each of these functions will not contain any logic for identification of the applicable tax rates or computation of the tax amount. This is where you would add your custom rules. The logic written within these functions is not limited to using just the parameters passed as part of the call and can fetch additional data points required to calculate taxes, as needed. Any failures in calculation logic in any of these functions should be configured to return an output error message. This error message must be added to the Merchandising Error Messages table or use an existing error message from that table. See the *Oracle Retail Merchandising Implementation Guide* section on Error Messages for details on how to add new error messages.

Custom Conflict Checking Rules

Within the Pricing solution, there is a process that occurs when price changes and clearance events are approved that validates that the change will not result in an invalid price at the location when it goes into effect. For example, that a combination of future price changes does not drive the price negative. This checking can also occur when a price event is unapproved, or when it is submitted (based on a system option). The details on the base conflict checking rules are outlined in the *Oracle Retail Pricing Implementation Guide*.

For **on premise implementations**, you also have the ability to add custom conflict checking rules, if you have additional conditions that you want to validate. Any custom rules are executed only after all the base rules are validated.

Writing Custom Rule Functions

Custom rules must be written in a PL/SQL package function that follows a specific format. The function should take one input parameter of type VARCHAR2 and one output parameter that is a CONFLICT_CHECK_ERROR_TBL to hold any error or conflict information. The function should return 0 for failure and 1 for success.

Configure Rule in Pricing

After you have created your function, you will need to configure it into Pricing. First, add your function so that it can be executed in the conflict processing. To do this you will need to add a row in the Conflict Query Control table in Pricing (RPM_CONFLICT_QUERY_CONTROL). The data that should be entered in the table is as follows:

Table Columns	Description
Query Control ID	A unique ID for the rule
Function Name	The full package function name in the package_name.function_name format

Table Columns	Description
Description	A functional description of the rule
Active	Use this value to indicate whether or not the rule is currently active. If a rule later is determined to be not needed, it can be updated to N. Otherwise, this should be Y. Valid values are Y or N.
Override Allowed	This field is not used by Pricing. Default to either Y or N.
Execution Order	This will allow you to determine what order you want your custom rules to execute, if you have more than one.
Base Generated	This should always be N for custom rules.
Base Required	This should always be N for custom rules.

Here is an example of how the insert could be created:

```
insert into RPM_CONFLICT_QUERY_CONTROL (
    CONFLICT_QUERY_CONTROL_ID,
    CONFLICT_QUERY_FUNCTION_NAME,
    CONFLICT_QUERY_DESC,
    ACTIVE,
    OVERRIDE_ALLOWED,
    EXECUTION_ORDER,
    BASE_GENERATED,
    BASE_REQUIRED)
select RPM_CONFLICT_QUERY_CONTROL_SEQ.nextval,
       'CUSTOM_RULE.BELOW_FIVE_CHECK',
       '$5 check',
       'Y',
       'Y',
       20,
       'N',
       'N'
from dual;
```

Then, define the error string you used in your package function in the RAF_RESOURCE_BASE table, similar to the following:

```
insert into raf_resource_base (app_code,
                             bundle_id,
                             locale_code,
                             resource_key,
                             source_text,
                             token_exists,
                             app_version)
select 'Rpm',
       bundle_id,
       'en',
       'below_five_check_string',
       'No item should sell for less than $5.',
       'N',
       app_version
from raf_resource_bundle
where bundle_text = 'oracle.retail.apps.rpm.model.RpmModelBundle'
and app_code = 'Rpm';
```

Example

Below is an example of a how to write a custom rule. The rule in this case is validating that no price is set below 5 USD.

```

CREATE OR REPLACE PACKAGE BODY CUSTOM_RULE AS
-----
FUNCTION BELOW_FIVE_CHECK(IO_error_table      IN OUT CONFLICT_CHECK_ERROR_TBL)
                        I_price_event_type IN   VARCHAR2)
RETURN NUMBER IS

    L_program VARCHAR2(61) := 'CUSTOM_RULE.VALIDATE'

    L_error_rec CONFLICT_CHECK_ERROR_REC := NULL;
    L_error_tbl CONFLICT_CHECK_ERROR_TBL := CONFLICT_CHECK_ERROR_TBL();

    cursor C_CHECK is
        select price_event_id,
               future_retail_id
        from rpm_future_retail_gtt
        where price_event_id NOT IN (select ccet.price_event_id
                                     from table(cast(L_error_tbl as
conflict_check_error_tbl)) ccet)
        and (   selling_retail      < 5
              or clear_retail      < 5);

BEGIN

    if IO_error_table is NOT NULL and
       IO_error_table.COUNT > 0 then
        L_error_tbl := IO_error_table;
    else
        L_error_rec := CONFLICT_CHECK_ERROR_REC(-99999, NULL, NULL, NULL);
        L_error_tbl := CONFLICT_CHECK_ERROR_TBL(L_error_rec);
    end if;

    for rec IN C_CHECK loop
        L_error_rec := CONFLICT_CHECK_ERROR_REC(rec.price_event_id,
                                                rec.future_retail_id,
                                                RPM_CONFLICT_LIBRARY.CONFLICT_ERROR,
                                                'below_five_check_string');

        if IO_error_table is NULL then
            IO_error_table := CONFLICT_CHECK_ERROR_TBL(L_error_rec);
        else
            IO_error_table.EXTEND;
            IO_error_table(IO_error_table.COUNT) := L_error_rec;
        end if;
    end loop;

    return 1;

EXCEPTION
    when OTHERS then
        L_error_rec := CONFLICT_CHECK_ERROR_REC(NULL,
                                                NULL,
                                                RPM_CONSTANTS.PLSQL_ERROR,
                                                SQL_LIB.CREATE_MSG('PACKAGE_ERROR',
                                                                    SQLERRM,
                                                                    L_program,
                                                                    TO_CHAR(SQLCODE)));

        IO_error_table := CONFLICT_CHECK_ERROR_TBL(L_error_rec);

```

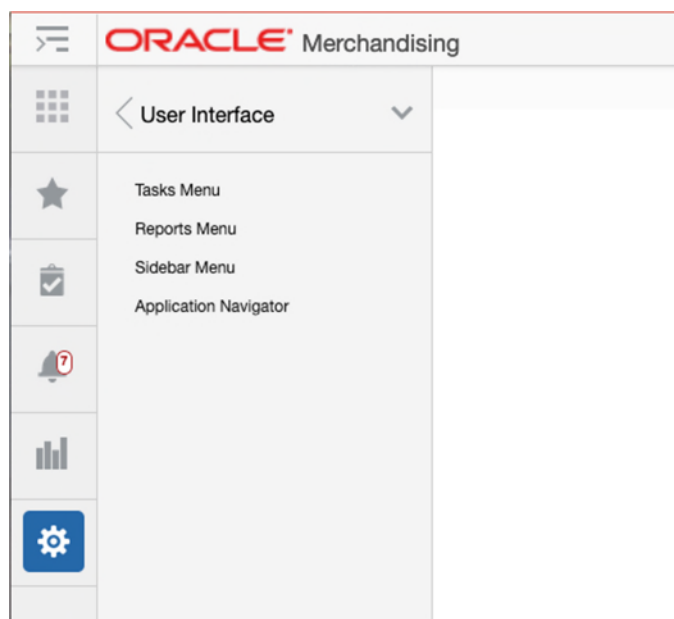
```
    return 0;  
END BELOW_FIVE_CHECK;  
----
```

Application Customization

The Merchandising Suite provides different options for customizing the user interface to meet your business practices, and in some cases to surface custom content. These options are accessed from the Settings in the sidebar menu in each of the Merchandising solutions.

User Interface

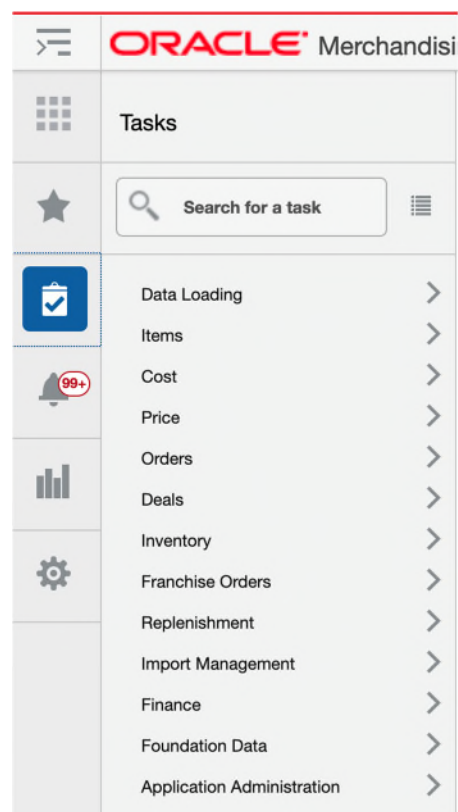
User interface customization is an option that can be used to customize the look and feel of the tasks menu, reports menu, sidebar menu and Application Navigator in the Merchandising solutions.



Tasks

The Tasks Menu link opens the Tasks Menu page, which allows an administrator can add, modify or remove the tasks available. For example, you may want to add a link for your users for another solution that is commonly used or a bolt-on application.

Tasks Menu



The Task Menu page displays buttons to save, restore or cancel the changes. When changes are saved, they will be immediately reflected in the Tasks menu. The **Cancel** button closes the tab without saving the changes. The **Restore** button replaces the customized menu with the with the base product configuration.

You can also click on the **View XML** button to view the XML version of the menu in an XML editor. The XML view contains a **View Tables** button which takes the user back to the tables view. The XML view also has a **View Revisions** button that allows you to view prior revisions of the task list. This should be used to re-apply any customizations you make after a patch is applied, as customizations are not retained during patching.

Table View

The screenshot shows the Oracle Pricing application interface. The main window displays a task menu with the following items:

- Price Inquiry
- Data Loading (selected)
- Price Changes
- Promotions
- Clearances
- Foundation Data
- Application Administration

The 'Attributes' table for the selected 'Data Loading' task is as follows:

Attribute	Value	Parameters
URL		
ID	DataLoading	
Visible	{securityContext.userInRole['VIEW_DATA_LOADING_	
Type	folder	
Title	Data Loading	
Target Title		
Target		
Show Close Icon		
Default Content		
Open Edit		
Contextual Area Width		
Contextual Area Collapsed		
Reuse Instance		

Buttons at the bottom right: Restore, Save, Cancel.

XML View

The screenshot shows the XML View of the Task Menu model. The XML code is as follows:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <NavigationDefinition xmlns="urn:www.oracle.com/oracle.retail.apps.framework.navigation" id="Folder_1">
3   <Items>
4     <Item id="PriceInquiry" visible="{securityContext.userInRole['VIEW_PRICE_INQUIRY_PRIV']} and RpmApplicationConfigBean.notSimplifiedPricing}" type="taskflow"
5       <url>/WEB-INF/oracle/retail/apps/rpm/priceinquiry/search/Flow/PriceInquirySearchFlow.xml#PriceInquirySearchFlow</url>
6       <Parameters>
7         <Parameter id="ignoreSaveWarning">true</Parameter>
8       </Parameters>
9     </Item>
10    <Item id="DataLoading" visible="{securityContext.userInRole['VIEW_DATA_LOADING_STATUS_PRIV']} and RpmApplicationConfigBean.notSimplifiedPricing}" type="fol
11      <Items>
12        <Item id="InductionStatus" visible="{securityContext.userInRole['VIEW_DATA_LOADING_STATUS_PRIV']} and RpmApplicationConfigBean.notSimplifiedPricing}"
13          <url>/WEB-INF/oracle/retail/apps/merchcommons/viewcontroller/induction/status/Flow/InductionStatusFlow.xml#InductionStatusFlow</url>
14          <Parameters>
15            <Parameter id="viewControllerWorkflowHandlerClassName">oracle.retail.apps.rpm.viewcontroller.induction.status.api.RpmInductionStatusViewControl
16              <Parameter id="modelWorkflowHandlerClassName">oracle.retail.apps.rpm.model.induction.status.api.RpmInductionStatusModelWorkflowHandler</Paramete
17            </Parameters>
18          </Item>
19          <Item id="InductionDownloadBlankTemplate" visible="{securityContext.userInRole['VIEW_CLEARANCES_PRIV']} or securityContext.userInRole['VIEW_PRICE_C
20            <url>/WEB-INF/oracle/retail/apps/merchcommons/viewcontroller/induction/template/Flow/InductionDownloadBlankTemplateFlow.xml#InductionDownloadBlank
21            <Parameters>
22              <Parameter id="viewControllerWorkflowHandlerClassName">oracle.retail.apps.rpm.viewcontroller.induction.template.api.RpmInductionDownloadBlankTe
23              <Parameter id="modelWorkflowHandlerClassName">oracle.retail.apps.rpm.model.induction.template.api.RpmInductionDownloadBlankTemplateModelWorkFlo
24            </Parameters>
25          </Item>
26        </Items>
27      </Item>

```

Buttons at the bottom right: Restore, Save, Cancel.

The task menu model xml file uses the xml schema definition called NavigationModel.xsd for xml validation. The Navigation Model schema definition file is used to validate any changes done by the administrator on the task menu xml file. The task menu model file consists of a hierarchy of item elements. Each element represents a menu item in the task menu. The item element can either be a folder or has a URL, list of attributes and parameters that identify the task menu item.

Task Menu Model XML Items

An item in the task menu model renders as either a launchable link or a submenu in the tasks menu of the Merchandising solutions. A new item can be added for a new link or submenu. An existing Item can be modified to change an existing link or submenu. Below is the example of a task menu model Item.

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition ... >
  <Items>
    <Item id="myFolder" title="My Folder" type="folder">
      <Items>
        <Item id="myContent" type="taskflow" title="My Content">

```

```

        <url>
            /WEB-
INF/mycompany/CustomDashboardTaskFlow.xml#CustomDashboardTaskFlow
        </url>
        <Parameters>
            <Parameter id="productName">"Acme"</Parameter>
        </Parameters>
    </Item>
</Items>
</Item>
</Items>
</NavigationDefinition>

```

Task Menu Model Item Attributes

The following attributes apply to an Item element in the task menu model xml file.

Attribute	Description
visible	Indicates if the item should be rendered (visible) or not. It can be an EL Expression that evaluates to true or false. Or it can be a plain string value equal to "true" or "false". Defaults to "true".
type	Indicates the type of the item. The main values are "folder", "taskflow", "link". The type "folder" indicates that the item contains additional sub-navigation items. The type "taskflow" indicates it is a task flow, and the type "link" is used for URLs. For SaaS implementations, the type "taskflow" is not supported.
title	The attribute title is used to provide the title of the sidebar item. This attribute is a required attribute.
targetTitle	The attribute targetTitle is used to provide the title of the tab in the content area. This attribute is an optional attribute. If not provided, the value of the title attribute will be used as the title of the tab in the content area.
accessKey	The attribute accessKey is used to specify the keyboard keystroke or a group of keystrokes that are used to access the navigation item using the keyboard.
shortDesc	The attribute shortDesc is used to provide the description of the navigation item which will be displayed when the user hovers over a menu item.
showCloseIcon	Mainly used when the content is being rendered in a popup, and indicates if the 'x' icon should show or not on the top right corner of the popup to facilitate manual closing of the popup. Takes the values "true" or "false". Defaults to "false".
defaultContent	Indicates whether the corresponding items from the model show up by default in tabs in the content area when the page template loads. Takes values "true" or "false", or may even take an EL Expression evaluating to "true" or "false".

Attribute	Description
reuseInstance	The attribute reuseInstance is used to specify whether the navigation items with the same title and url will use the same tab or not. When this attribute is set to false, the request to open the content for that navigation item will always use a new tab. When the attribute is true, the navigation items with the same title and url will share the same tab in the content area. It defaults to true.
keysList	Takes name value pairs separated by a semicolon. The attribute keysList provides a way to differentiate two navigation items with the same title and url. If keysList is not provided, then the two navigation items with the same title and url will always use the same tab. When the keysList is provided, it provides uniqueness to the navigation items with the same title and url enabling them to use different tabs. Example keysList="key1=value1;key2=value2"
urlRendererHeight, urlRendererWidth	Example values are shown below. These used in the case of a popup and indicate the height and width of the popup. urlRendererHeight="200px" urlRendererWidth="200px"
reloadTab	When this attribute is set to true, an already opened tab will be reloaded with the new input parameters for the taskflow. When it is set to false, a previously opened tab will only be re-focused, but not reloaded with new input parameters for the taskflow.
refreshOnDisclosure	When the navigation item has refreshOnDisclosure attribute set to true then the tab displaying that item in the Content Area will be refreshed every time it's disclosed. The attribute can take either of the two values true or false. Default is false. The attribute is useful in the scenarios where we want to display to the user the latest information from the database every time he/she comes back to the tab. The attribute should be used with caution because if the data in that tab is not committed before leaving the tab then the uncommitted data will be lost upon coming back to the tab.
dynamicTabFocus	When a navigation item is invoked, the tab displaying that item will have its text focused. To override this behavior, set dynamicTabFocus to "false". This attribute defaults to "true".
popupId	Applicable only when target="_popup". Must be a number between 1 and 15. This attribute allows consuming applications to target a specific popup within the UI Shell. The framework provides 15 popups that consuming applications can take advantage of. In case this attribute has not been specified, a default popup will be used by the framework. This default popup will not store its dimensions in MDS.

Attribute	Description
popupContentHeight	Applicable only when target="_popup". This attribute is used to provide the height in pixels of the resulting popup dialog window.
popupContentWidth	Applicable only when target="_popup". This attribute is used to provide the width in pixels for the resulting popup dialog window.
popupStretchChildren	Applicable only when target="_popup". This attribute is used to indicate the stretching behavior for the contents of the resulting popup window. The contents are referred to as child components. Valid values are: <ul style="list-style-type: none"> ▪ none - does not attempt to stretch any children (the default value and value you need to use if you have more than a single child; also, the value you need to use if the child does not support being stretched). ▪ first - stretches the first child (not to be used if you have multiple children as such usage will produce unreliable results; also, not to be used if the child does not support being stretched).
popupResize	Applicable only when target="_popup". This attribute is used to indicate the resulting popup window's resizing behavior. Valid values are: <ul style="list-style-type: none"> ▪ on - user can resize the dialog with their mouse by dragging any of the dialog edges. ▪ off - the dialog automatically sizes its contents if popupStretchChildren is set to "none"
popupHelpTopicId	Applicable only when target="_popup". This attribute is used to look up a topic in a help-Provider for the resulting popup window.
popupShortDesc	Applicable only when target="_popup". This attribute is used to provide short description of the resulting popup window.
contentListener	Specifies the fully qualified name of the class implementing the ContentListener interface. This allows applications to have the ability to inject any session or request values before opening tabs.
tabShortDesc	Specifies the text to be shown when the user hovers over the application tab. Using this attribute application team can keep the title short and the tabShortDesc as fully qualified tab name which can be shown as the tooltip of the tab. This attribute will be displayed as tab title in screen reader mode.

Task Menu Model Item Sub-elements

The following are the sub-elements of the Item element in the task menu model.

Table 4–2 Task Menu Model Item Sub-elements

Element	Description
url	<p>The location of the item being launched.</p> <p>If the type is "taskflow" - then the URL element must contain the path to the task flow XML.</p> <p>If the type is "link" - then the URL of the external system must be indicated in this sub-element. The entire URL must be marked as character data (such as, enclosed in CDATA).</p>
Parameter	<p>The <Parameters> sub-element within <Item> should list all the parameters to the dashboard page if there are any. Each parameter is represented as a <Parameter> element inside <Parameters>.</p> <p>The <Parameter> id should be the actual parameter reference name recognized by the dashboard URL.</p> <p>The value of each <Parameter> is a string value. This is the only supported data type.</p>

Securing Access to Items

To restrict access to menu items to specific security roles, set the visible property on the <Item> element for the URL to an Expression Language (EL) expression that calls ADF's securityContext API's isUserInRole method. Example:

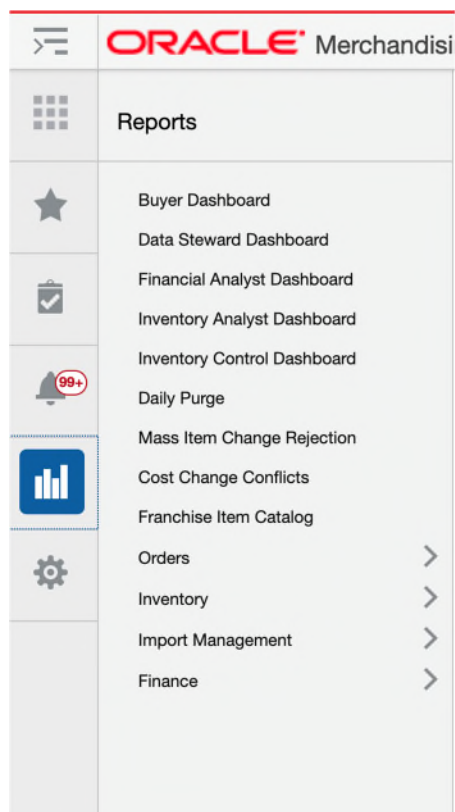
```
<Item id="myDashboard1"
      type="link"
      title="Profitability Dashboard"
      visible="#{securityContext.isUserInRole['BUYER_JOB']} ">
```

The parameter to the securityContext.isUserInRole method is a logical security role that is configured for the solution. The API returns true if the user is included in the specified security role. If the user is not authenticated or is not found in the role, the API returns false.

Reports Menu

The Reports Menu link opens a page that allows you to add, modify or remove the reports available in the Reports menu. For example, if you have built some custom reports using BI Publisher, you might add the links using this function.

Reports Menu



The Reports Menu page displays buttons to save, restore or cancel the changes. When changes are saved, they will be immediately reflected in the Reports menu. The **Cancel** button closes the tab without saving the changes. The **Restore** button replaces the customized menu with the original menu shipped with the Merchandising solution.

You can also click on the **View XML** button to view the XML version of the menu in an XML editor. The XML view contains a **View Tables** button which takes the user back to the tables view. The XML view also has a **View Revisions** button that allows you to view prior revisions of the Reports list. This should be used to re-apply any customizations you make after a patch is applied, as customizations are not retained during patching.

Table Views

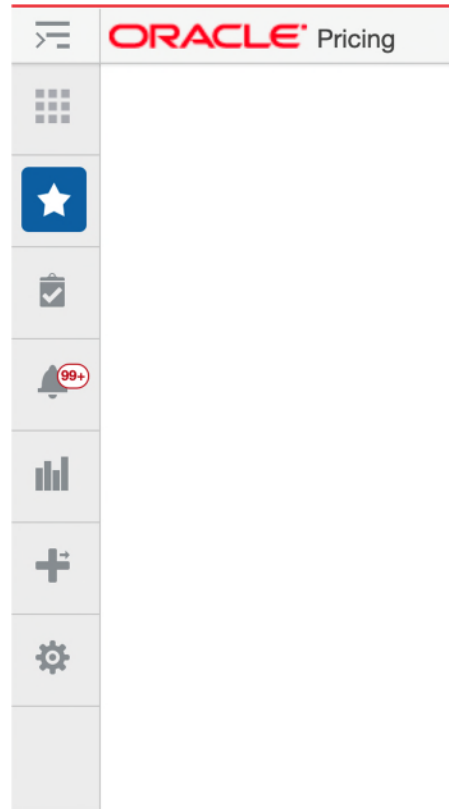
XML View

The Reports menu uses the same NavigationModel.xsd file used by the Task menu. Please refer to the Task Menu section above for the details on how to customize the reports menu. For SaaS implementations, only the type “link” is supported.

Sidebar Menu

The Sidebar Menu link opens a page that allows you to add, modify or remove the icons available in the sidebar menu for Merchandising solutions. For example, you could hide the Quick Create option in Allocation or Pricing, if you are not using that functionality.

Sidebar Menu



The Sidebar Menu page displays buttons to save, restore or cancel the changes. When changes are saved, they will be immediately reflected in the Sidebar menu. The **Cancel** button closes the tab without saving the changes. The **Restore** button replaces the customized menu with the with the base product configuration.

You can also click on the **View XML** button to view the XML version of the menu in an XML editor. The XML view contains a **View Tables** button which takes the user back to the tables view. The XML view also has a **View Revisions** button that allows you to view prior revisions of the Sidebar list. This should be used to re-apply any customizations you make after a patch is applied, as customizations are not retained during patching.

Table Views

The screenshot shows the Oracle Pricing application interface. The top navigation bar includes the Oracle logo and 'Pricing' text. The user is logged in as 'PRICING_ADMIN'. The sidebar menu on the left has a 'Quick Create' item selected. The main content area is divided into two sections:

- Menu Item:** A table with one row containing the text 'Quick Create'.
- Attributes:** A table with columns 'Attribute', 'Value', and 'Parameters'. The attributes listed are:

Attribute	Value	Parameters
Taskflow	/WEB-INF/oracle/retail/apps/rpn/pricchange/quickcreate	View
ID	quickCreateSidebarItem	
Title	Quick Create	
Visible	<input type="checkbox"/>	
Icon Uri	/raf/retail-core/images/quick-create-gray.png	
Selected icon Uri	/raf/retail-core/images/quick-create.png	
Disabled icon Uri	/raf/retail-core/images/quick-create-gray.png	
Hover icon Uri	/raf/retail-core/images/quick-create-gray.png	
Depressed icon Uri	/raf/retail-core/images/quick-create-gray.png	

At the bottom right of the main area, there are buttons for 'Restore', 'Save', and 'Cancel'.

XML View

The screenshot shows the Oracle Pricing application interface in XML view. The top navigation bar and sidebar menu are the same as in the previous screenshot. The main content area displays the XML code for the sidebar menu item:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <SidebarModel xmlns="urn:www.oracle.com:oracle.retail.apps.framework.sidebar" id="raf_sidebar_items_root">
3 <Items>
4 <item id="quickCreateSidebarItem" iconUri="/raf/retail-core/images/quick-create-gray.png" selectedIconUri="/raf/retail-core/images/quick-create.png" title=
5 <taskflow>/WEB-INF/oracle/retail/apps/rpn/pricchange/quickcreate/flow/PriceEventQuickCreateFlow.xml#PriceEventQuickCreateFlow/taskflow</
6 <Parameters>
7 <Parameter id="menuName">Quick Create</Parameter>
8 </Parameters>
9 </Item>
10 </Items>
11 </SidebarModel>
12
  
```

At the bottom right of the main area, there are buttons for 'Restore', 'Save', and 'Cancel'.

The sidebar menu uses a schema called sidebarSchema.xsd for the validation of the sidebar items. The sidebar xml model file consists of item elements similar to the Item element in the Task menu, but the item element in the sidebar xml model has a different set of attributes.

Sidebar Menu Model XML Items

An item element in the sidebar model xml file renders as a launchable icon in the sidebar. The sidebar has a fixed set of icons and a variable set of icons. Only the variable icons can be customized. Below is the example of a sidebar menu model Item.

```
<?xml version="1.0" encoding="UTF-8" ?>
<SidebarModel id="raf_sidebar_items_root"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="urn:www.oracle.com:oracle.retail.apps.framework.sidebar
classpath:oracle/retail/apps/framework/uishell/common/sidebar/sidebarSchema.xsd"
xmlns="urn:www.oracle.com:oracle.retail.apps.framework.sidebar">
  <Items>
<Item id="quickCreateSidebarItem" title="Quick Create" iconUrl="/raf/retail-
core/images/quick-create-gray.png" selectedIconUrl="/raf/retail-core/images/quick-
create.png">

<taskFlow>/WEB-INF/oracle/retail/apps/framework/uishell/navigation/contentarea/flo
w/TestQuickCreateFlow.xml#TestQuickCreateFlow</taskFlow>
</Item
  </Items>
</SidebarModel>
```

Sidebar Menu Model Item Attributes

The following attributes apply to an Item element in the sidebar menu model xml file.

Sidebar Menu Model Item Attributes

Attribute	Description
id	The id of the sidebar item.
title	The title of the sidebar icon.
visible	The visible attribute controls the visibility of the sidebar item.
iconUrl	The icon that is displayed for the sidebar item.
selectedIconUrl	The icon that is displayed when the sidebar item is selected.
disabledIconUrl	The icon that is displayed when the sidebar item is disabled.
hoverIconUrl	The icon that is displayed when the sidebar item is hovered on.
depressedIconUrl	The icon that is displayed when the sidebar item is depressed.

Sidebar Model Item Sub-elements

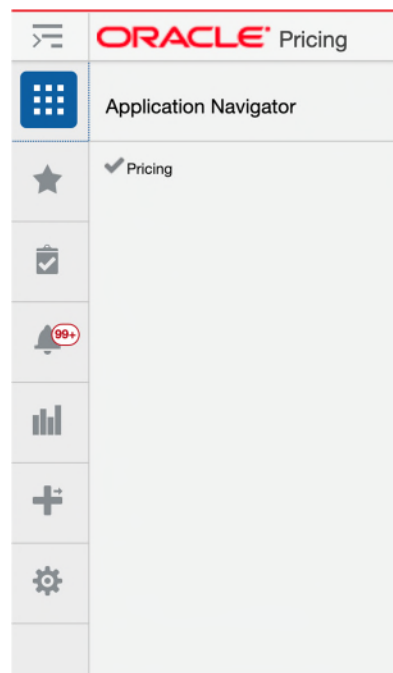
The following are the sub-elements of the Item element in the sidebar model.

Sidebar Model Item Sub-elements

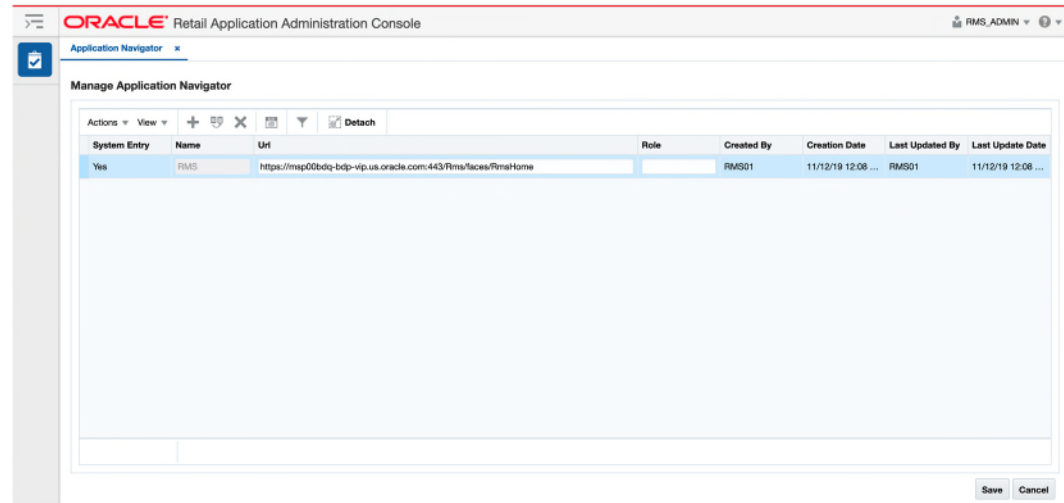
Element	Description
taskFlow	The URL of the taskflow being launched.
Parameter	<p>The <Parameters> sub-element within <Item> should list all the parameters to the sidebar taskflow if there are any. Each parameter is represented as a <Parameter> element inside <Parameters>.</p> <p>The <Parameter> id should be the actual parameter reference name recognized by the taskflow URL.</p> <p>The value of each <Parameter> is a string value. This is the only supported data type.</p>

Application Navigator

Application Navigator allows users to launch different applications from the Merchandising solution they are currently using so that they can shuffle between multiple applications based on their privileges and avoid having to open a new tab and enter a new URL to launch an application. Users can instead click on the solution from a list, which will launch that application in a new tab or window based on the browser settings.



If a user has access to multiple applications (based on their defined role) you can configure their list to show all available solutions in the Application Navigator page. Otherwise only the current solution will be displayed. Clicking on the Application Navigator option in the Settings > User Interface menu will open the Application Navigator screen in the Oracle Retail Application Administrator Console.



Add

The Add action is enabled at all times and allows an administrator to add a new URL.

1. From the Actions menu, select Add. A new, empty entry is added to the table.
2. Enter a name, the application URL, and the role to which the URL applies. The remaining columns are populated automatically.
3. Click Save to save your changes.

Modify

While all the columns on a row can be modified, only the Role and URL columns of a System record can be modified by the administrators.

Duplicate

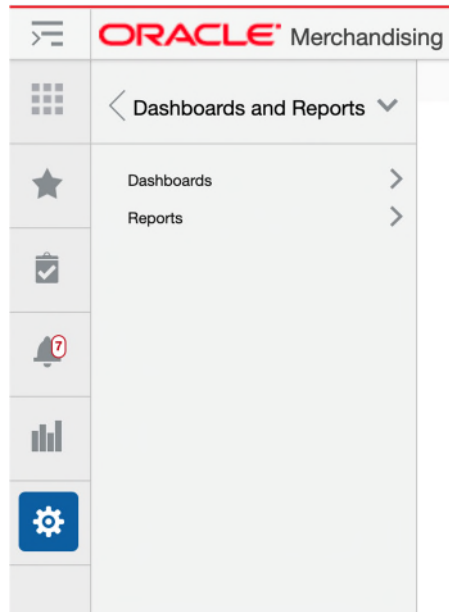
The Duplicate action adds a copy of each selected row below its originating row. The object name or unique identifier of the duplicate row is "Copy of <Object Name>" and appears in edit mode to prevent accidentally creating duplicate labels. This function may be helpful if you want to add the same link for multiple roles.

Delete

The Delete action is enabled when an entry is selected. When the user selects an application navigator entry and clicks **Delete**, the user is prompted with a warning message. Click Yes and the selected entry is removed.

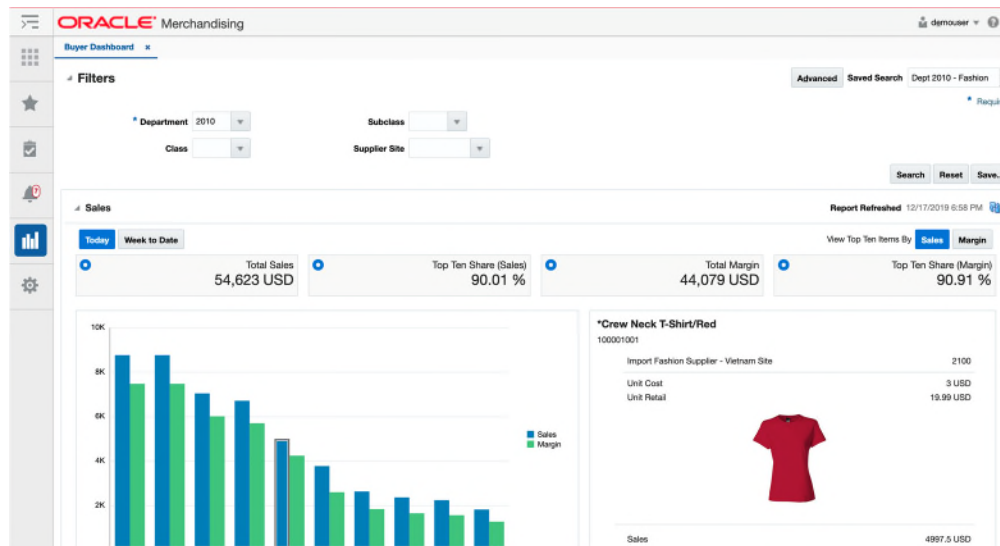
Dashboards and Reports

The Dashboard and Reports menu under Settings provides options to customize the reports displayed in Dashboards and the contextual pane in the Merchandising solutions.

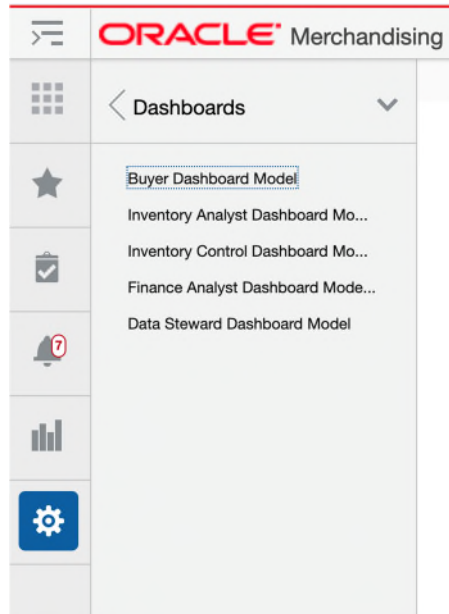


Dashboards

Dashboards are used by Merchandising solutions to surface exceptions and key metrics to users. Each dashboard is targeted for specific user roles, like the Buyer dashboard shown below. Most dashboards have two main sections – the filters and the reports. The base solutions come with pre-defined reports, which can be used. But, you can also add your own reports and remove ones that do not apply to your business. This is done through the Settings > Dashboards link.



Clicking on the Dashboards option will show you a list of the available dashboards in the Merchandising solution you are logged into. To modify the reports shown in the dashboard, click the link for the dashboard you wish to modify.



This will open the Dashboard Model page. In this page, you can add new reports, modify the appearance of existing reports, reorder the reports in the dashboard, or remove the reports from display.

When changes are saved, they will not be immediately applied in the updated dashboard if it is opened. It must be closed and re-opened in order for the changes to be reflected. The **Cancel** button closes the tab without saving the changes. The **Restore** button replaces the customized dashboard with the base product configuration.

You can also click on the **View XML** button to view the XML version of the dashboard in an XML editor. The XML view contains a **View Tables** button which takes the user back to the tables view. The XML view also has a **View Revisions** button that allows you to view prior revisions of the dashboard. This should be used to re-apply any customizations you make after a patch is applied, as customizations are not retained during patching.

When adding or modifying reports, the following parameters are applicable:

Name	Description	Required
id	A unique id for this region	Yes
type	The type of content contained in this region. Valid values: prompt, report	Yes
width	The width of this region. See below for more details.	No
height	The height of a region. See below for more details.	No
url	The task flow URL to open in this region	Yes
Parameters	Parameters to be sent to the URL	No

Table View

Oracle Merchandising - Buyer Dashboard Model

Dashboard Orientation: Rows Columns

Layout Item:

- Header
 - Column 1
 - Report: DailySales
 - Report: EarlyLateShipments
 - Report: OpenToBuy
 - Report: OrdersToApprove

Header Configuration:

Attribute	Value	Parameters
URL	/WEB-INF/oracle/retail/apps/framework/dvtcontextaw	View
Visible		
Width	100%	
Height	auto	
Item Type	report	
Report ID	OpenToBuy	

Restore Save Cancel

XML View

Oracle Merchandising - Buyer Dashboard Model

View Tables View Revisions

Find Go to Line

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <-dashboard xmlns="urn:www.oracle.com/oracle_retail_apps_framework_dashboard" layout="column">
3 <-HeaderItem id="filter" type="prompt" width="100%" height="auto">
4 <-url>/WEB-INF/oracle/retail/apps/rms/dashboards/view/buyer/flow/BuyerDashboardFilterFlow.xml#BuyerDashboardFilterFlow</url>
5 </HeaderItem>
6 <-Vector>
7 <-Vector width="100%" height="auto" dimensionsFrom="auto">
8 <-Items>
9 <-Item id="DailySales" type="report" width="100%" height="auto">
10 <-url>/WEB-INF/oracle/retail/apps/framework/dvtcontextawarereport/publicui/flow/DVTContextAwareReportFlow.xml#DVTContextAwareReportFlow</url>
11 <-Parameters>
12 <-Parameter id="taskFlowURL">/WEB-INF/oracle/retail/apps/rms/dashboards/view/buyer/flow/DailySalesReportFlow.xml#DailySalesReportFlow</Parameter>
13 <-Parameter id="parameterName1">departmentId</Parameter>
14 <-Parameter id="payloadKeyName1">department</Parameter>
15 <-Parameter id="parameterName2">classId</Parameter>
16 <-Parameter id="payloadKeyName2">classId</Parameter>
17 <-Parameter id="parameterName3">subclassId</Parameter>
18 <-Parameter id="payloadKeyName3">subclass</Parameter>
19 <-Parameter id="parameterName4">storeId</Parameter>
20 <-Parameter id="payloadKeyName4">store</Parameter>
21 <-Parameter id="parameterName5">supplierSiteId</Parameter>
22 <-Parameter id="payloadKeyName5">supplierSite</Parameter>
23 <-Parameter id="parameterName6">brands</Parameter>
24 <-Parameter id="payloadKeyName6">brandName</Parameter>
25 <-Parameter id="parameterName7">originCountries</Parameter>
26 <-Parameter id="payloadKeyName7">countryOfOrigin</Parameter>
27 <-Parameter id="parameterName8">sessionId</Parameter>

```

Restore Save Cancel

Adding or Replacing a Report

The following are the steps needed to add or replace a report on an included dashboard:

1. In the Settings Menu, click on the Dashboard folder and then click on the name of the dashboard you wish to edit.
2. Note the new report's task flow URL and parameters.
3. Modify the Dashboard Prompt Configuration XML file to add an item or replace an existing item for the new report.
4. If the report has to refresh when a prompt on the dashboard is changed by the user, make sure that report task flow is wrapped in a DVTContextAwareTaskFlow for its item entry in the dashboard prompt configuration XML file.
5. Test the Dashboard.

Removing a Report

The following are the steps needed to remove a report from an included dashboard:

1. In the Settings Menu, click on the Dashboard folder and then click on the name of the dashboard you wish to modify.

2. Modify the dashboard prompt configuration XML file and remove the report.
3. You may need to re-adjust the position of the other reports.
4. Test the Dashboard.

Change a Report Layout

The following are the steps needed to change the layout of the reports in included dashboards:

1. In the Settings Menu, click on the Dashboard folder and then click on the name of the dashboard you wish to modify.
2. Modify the dashboard prompt configuration XML file and rearrange the position of the item entries. Save the dashboard configuration file.
3. Re-open and test the Dashboard.

Finally, if you would like to configure a dashboard for users such that when they log into the Merchandising solution it is displayed automatically, you should configure the Default Content parameter for the privilege associated with that dashboard. For example, if you wanted your finance analysts to have the Finance Analyst dashboard open automatically when they log into Merchandising, you'd configure something like this into the Default content parameter:

```
#{securityContext.userInRole[ 'VIEW_FINANCE_ANALYST_DASHBOARD_PRIV' ]}
```

Reports

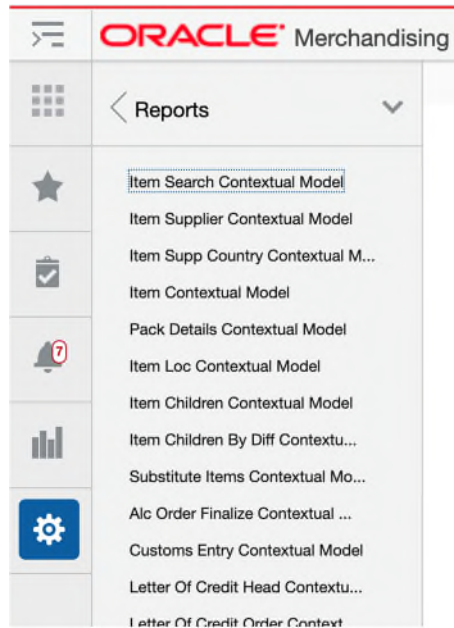
Contextual reports are available in many key workflows throughout the Merchandising Suite of solutions to provide additional information to the user of a particular workflow to help in decision making. This can also be an area of extensibility, as you are able to configure your own reports as well. This is done through the Settings > Reports link.

The screenshot displays the Oracle Merchandising application interface. The main area shows search results for 'Mid Rise Skinny Jean'. The results table includes columns for Item, Description, Item Level, Department, Class, Subclass, Subclass Name, and Status. The following table represents the data shown in the screenshot:

Item	Description	Item Level	Department	Class	Subclass	Subclass Name	Status
100108086	Mid Rise Skinny Jean	Level 1	2010	3	1	Fashion *	Approved
100108131	Mid Rise Skinny Jean:Antique:10	Level 2	2010	3	1	Fashion *	Approved
100108271	Mid Rise Skinny Jean:Classic Wash:10	Level 2	2010	3	1	Fashion *	Approved
100108174	Mid Rise Skinny Jean:Dark Wash:4	Level 2	2010	3	1	Fashion *	Approved

On the right side of the interface, there is an 'Item Details' pane for 'Mid Rise Skinny Jean', showing a product image and additional information such as 'Supplied by Fashion Supplier Domestic Site (2000)', 'Unit Cost', 'Unit Retail' (89.99 USD), 'Ownership', 'Origin Country' (US), and 'Pack Size' (20).

Clicking on the Reports option will show you a list of the available pages in the Merchandising solution you are logged into that have a contextual pane enabled for contextual reports. To add, modify, or remove reports shown in a page's contextual pane, click the link for the page you wish to modify.

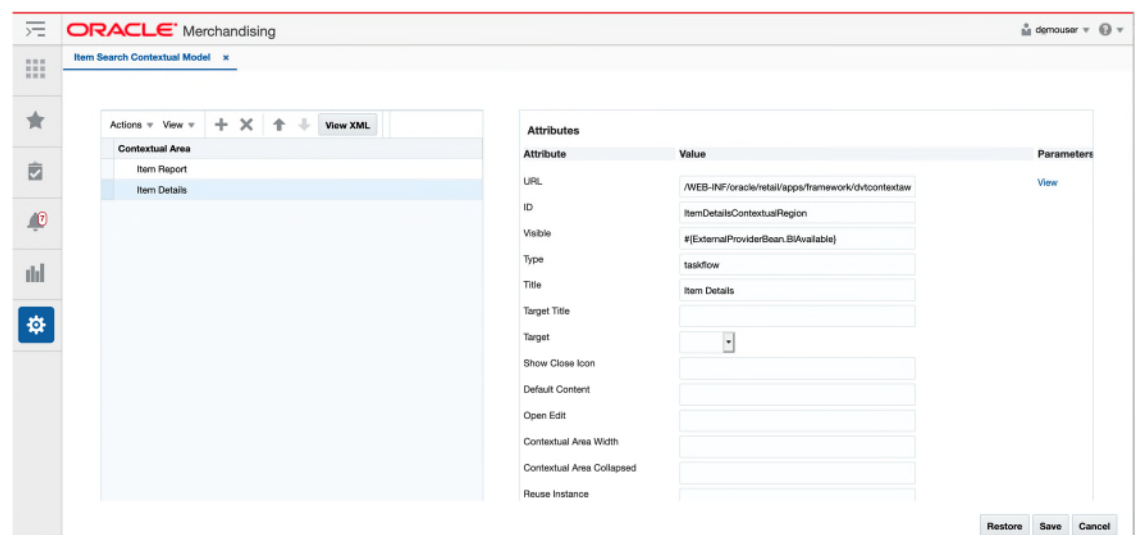


This will open the Contextual Model page. In this page, you can add new reports, modify the appearance of existing reports, reorder reports, or remove reports from display.

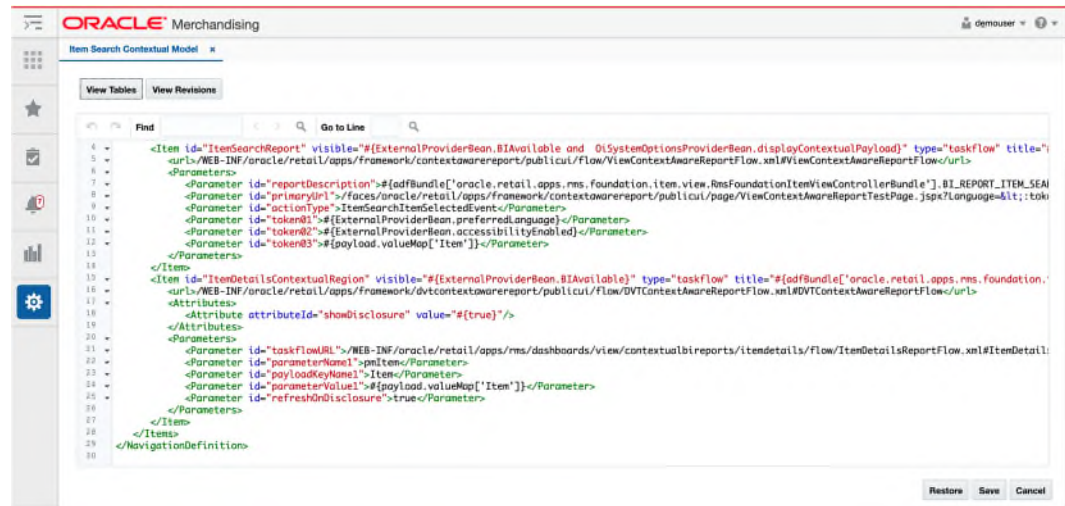
When changes are saved, they will not be immediately applied. The contextual pane in the impacted page must be closed and reopened to refresh the view. The **Cancel** button closes the tab without saving the changes. The **Restore** button replaces the customized dashboard with the base product configuration.

You can also click on the **View XML** button to view the XML version of the contextual pane in an XML editor. The XML view contains a **View Tables** button which takes the user back to the tables view. The XML view also has a **View Revisions** button that allows you to view prior revisions of the contextual pane. This should be used to re-apply any customizations you make after a patch is applied, as customizations are not retained during patching.

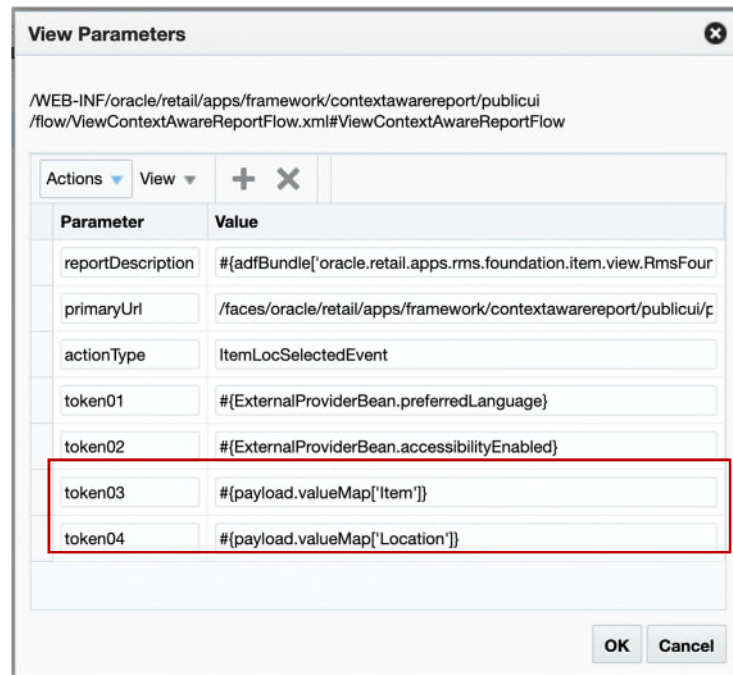
Table View



XML View



The reports are considered contextual, as they can use parameters that are published by the workflow in which it is displayed. The payload that is published by each of the workflows differs, so in order to configure other reports in a particular workflow, you will need to review what is available. This can be viewed in the Contextual Model page by clicking on the View link under Parameters in the table view. For example, the figure below shows that for the Item Location workflow in Merchandising, there are two parameters – Item and Location.



Adding a URL-based Contextual Report

To configure any non-ADF DVT report that is URL based into a contextual pane, follow these steps:

1. In the Settings Menu, click on the Reports folder and then select the model XML for the workflow you wish to configure. This will open the workflow's contextual area model.
2. Switch to View XML.
3. Add an <Item> element within the topmost <Items> element that references the task flow called ViewContextAwareReportFlow. The ViewContextAwareReportFlow is a framework for rendering URL based reports that will be aware of contextual business events emanating from the task flows in the Merchandising solutions.
4. Make sure that the <Item> id is unique. Make sure the <Item> type is "taskflow". Provide a meaningful title.

Populate the parameters to the ViewContextAwareReportFlow by adding the following <Parameter>/<Parameters> elements.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition ... >
  <Items>

    <Item id="showCustomerMetric" type="taskflow" title="Customer Metric">
      <url>
        /WEB-INF/oracle/retail/apps/framework/contextawarereport/publicui/flow/ViewContextA-
        wareReportFlow.xml#ViewContextAwareReportFlow
      </url>

      <Parameters>
        <Parameter id="reportDescription">Item Metric</Parameter>
        <Parameter id="actionType">AllocMaintItemSelectedEvent</Parameter>
        <Parameter id="primaryUrl">
          <![CDATA[/faces/oracle/retail/apps/framework/contextawarereport/publicui/page/V
          iewContextAwareReportTestPage.jspx?paramItemId=<selectedItemId>&paramItemType=<
          selectedItemType:token01>&paramLanguage=<language>]]>
        </Parameter>
        <Parameter id="token01">regular</Parameter>
      </Parameters>
    </Item>
  </Items>
</NavigationDefinition>
```

Note the following:

- The <Parameter id="reportDescription"> element is the title of the contextual area report. Set this to a meaningful value
- The <Parameter id="actionType"> element indicates the contextual business event name the report will listen to.
- The <Parameter id="primaryUrl"> element indicates the URL to the contextual area report in the BI server. The entire URL must be marked as character data (such as, enclosed in CDATA).

Note: that the parameters to the URL are tokenized. The example above uses a test page called ViewContextAwareReportTestPage.jspx, which can be replaced with the actual report URL.

- The "paramItemId=<selectedItemId>" portion of the URL instructs the system to pass the contextual business event payload value called selectedItemId into the URL parameter paramItemId when rendering the contextual report.

- The "paramItemType=<selectedItemType:token01>" portion of the URL instructs the system to pass the contextual business event payload value called `selectedItemType` into the URL parameter `paramItemType` when rendering the contextual report. If that payload value is empty or null at runtime, then a default value of `regular` is used as referenced in a `<Parameter id="token01">` entry. The colon symbol separates the payload value from the default value if the payload value is null.
 - The "paramLanguage=<language>" portion of the URL instructs the system about the current locale of the user. The "language" identifier is a reference to a value in the contextual event payload. This is a built-in value that all Retail Application contextual business event payloads will have.
 - The `<Parameter id="token01">` element holds the default value for the URL parameter `selectedItemType`. Token parameters hold default values and you can define up to 20 default value tokens.
5. Test the changes by going to the flow where the report was added and verifying that the report is rendered correctly.

Adding an ADF-DVT based Contextual Report

If the report you want to configure is an ADF DVT-based report¹, then the following steps should be followed:

1. In the Settings Menu, click on the Reports folder and then select the model XML for the workflow you wish to configure. This will open the workflow's contextual area model.
2. Switch to View XML.
3. Add an `<Item>` element within the topmost `<Items>` element that references the task flow called `DVTContextAwareReportFlow`. The `DVTContextAwareReportFlow` is a framework for rendering ADF DVT based reports that will be aware of contextual business events emanating from the task flows in the Merchandising solutions.
4. Make sure that the `<Item>` id is unique. Make sure the `<Item>` type is "taskflow". Provide a meaningful title.
5. Populate the parameters to the `DVTContextAwareReportFlow` by adding the following `<Parameter>/<Parameters>` elements.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition ... >
  <Items>

    <Item id="showCustomerMetric" type="taskflow" title="Customer Metric">
      <url>
        /WEB-INF/oracle/retail/apps/framework/dvtcontextawarereport/publicui/flow/DVTContextAwareReportFlow.xml#DVTContextAwareReportFlow
      </url>
      <Parameters>
        <Parameter
          id="taskflowURL">/WEB-INF/oracle/retail/apps/framework/uishell/navigation/contextualarea/flow/ItemMetricFlow.xml#ItemMetricFlow</Parameter>
        <Parameter id="parameterName1">itemId</Parameter>
        <Parameter id="payloadKeyName1">itemId</Parameter>
```

¹ For non-SaaS implementations only

```

<Parameter id="parameterValue1">#{payload.valueMap['ItemId']}</Parameter>
<Parameter id="refreshOnDisclosure">#{true}</Parameter>
</Parameters>
</Item>
</Items>
</NavigationDefinition>

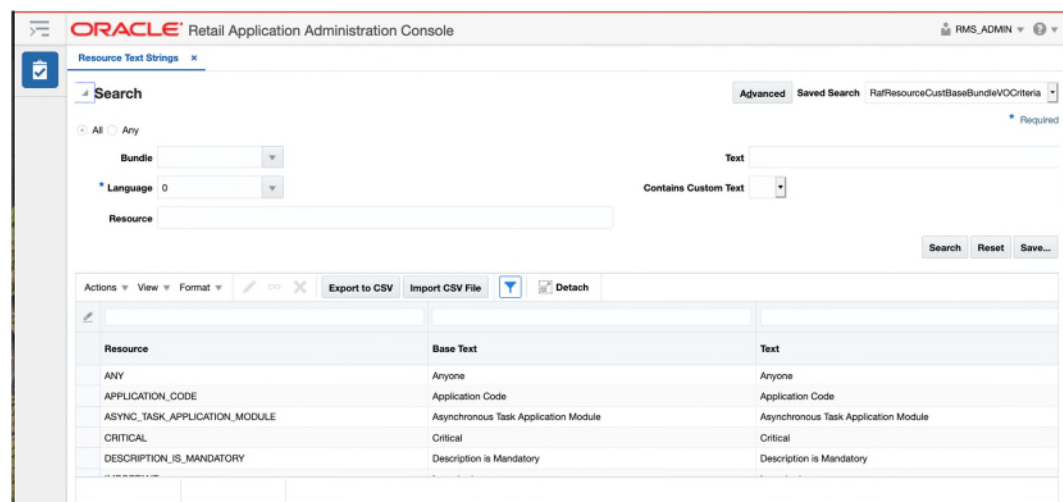
```


Note the following:

- The <Parameter id="taskflowURL"> element is the URL of the DVT taskflow that will be displayed in the contextual area.
 - The <Parameter id="parameterName1"> element indicates a parameter for the DVT taskflow. A maximum of 15 parameters can be specified.
 - The <Parameter id="payloadKeyName1"> element indicates the name of the payload key that maps to the taskflow parameter parameterName1. A maximum of 15 payload keys can be specified. The value of the payload key in the event payload will be used for the parameter in the DVT taskflow.
 - The <Parameter id="parameterValue1"> element holds the default value of the DVT taskflow. A maximum of 15 parameter values can be specified.
 - The <Parameter id="refreshOnDisclosure"> element is used to refresh the DVT taskflow on the disclosure of the tab.
6. Test the Retail Application. Go to the flow where the report was added and verify that the report is rendered correctly.

Resource Bundles

The Resource Bundles option in the customization menu has screens that allow you to customize certain labels, error messages, and list of values used by Merchandising Solutions. The Resource Text Strings option provides a screen to do this customization for your primary language or other supported languages. It can also be used for adding translations for languages that are not supported in the Merchandising solutions.



To update the description, select the language in which you want to make your updates and optionally a partial text string to search for the instances you want to update. Then, click the Edit option in the Actions menu or the iconic button ().

This will open a popup where you can enter your custom text. If there are a lot of strings that require updates, you can also select to export the data to a CSV and then re-import once your changes have been applied.

The other option under Resource Bundles, entitled Imports Management, allows you to view the status of the imported files and any errors that occurred during the re-import process. This screen also allows you to import files and re-process that previously encountered errors, if you believe the issue has been corrected.

Status	File Name	Errors	Created By	Creation Date	Last Updated By	Last Update Date
✓	export.csv	0	RMS_ADMIN	13/9/19 10:54 AM	RMS_ADMIN	13/9/19 10:54 AM
✓	export.csv	0	RMS_ADMIN	13/9/19 10:23 AM	RMS_ADMIN	13/9/19 10:23 AM
✓	export.csv	0	RMS_ADMIN	13/9/19 10:21 AM	RMS_ADMIN	13/9/19 10:21 AM
✓	export.csv	0	RMS_ADMIN	13/9/19 10:20 AM	RMS_ADMIN	13/9/19 10:20 AM
⚠	export.csv	15 (View)	RMS_ADMIN	13/9/19 10:18 AM	RMS_ADMIN	13/9/19 10:18 AM
⚠	export.csv	15 (View)	RMS_ADMIN	13/9/19 10:16 AM	RMS_ADMIN	13/9/19 10:16 AM
✓	export.csv	0	RMS_ADMIN	13/9/19 10:11 AM	RMS_ADMIN	13/9/19 10:11 AM
⚠	oracle.retail.apps.rms.inventory.mo...	15 (View)	RMS_ADMIN	13/9/19 10:02 AM	RMS_ADMIN	13/9/19 10:02 AM
⚠	115_oracle.retail.apps.rms.inventor...	15 (View)	RMS_ADMIN	13/9/19 10:01 AM	RMS_ADMIN	13/9/19 10:01 AM
⚠	115_oracle.retail.apps.rms.inventor...	15 (View)	RMS_ADMIN	13/9/19 9:59 AM	RMS_ADMIN	13/9/19 9:59 AM
⚠	115_oracle.retail.apps.rms.inventor...	15 (View)	RMS_ADMIN	13/9/19 9:51 AM	RMS_ADMIN	13/9/19 9:51 AM
⚠	115_oracle.retail.apps.rms.inventor...	15 (View)	RMS_ADMIN	13/9/19 9:47 AM	RMS_ADMIN	13/9/19 9:47 AM

Note: Merchandising also holds a number of labels and error messages in database tables. For information on updating these, see the Codes and Descriptions and Error Messages sections of the *Oracle Retail Merchandising Implementation Guide*.