Oracle® E-Business Suite

Extending Enterprise Command Centers

Release 12.2

Part No. F21671-11

April 2024

ORACLE

Oracle E-Business Suite Extending Enterprise Command Centers, Release 12.2

Part No. F21671-11

Primary Author: Nashwa Ghazaly, Muhannad Obeidat

Contributing Author: Mildred Wang

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 customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Contents

8 How to Extend Oracle Enterprise Command Centers

9 Personalization Template

10 Implementing the Push Model for Loading Data

A Sample PL/SQL Package

Send Us Your Comments

Oracle E-Business Suite Extending Enterprise Command Centers, Release 12.2

Part No. F21671-11

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 new Oracle E-Business Suite Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_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

Intended Audience

Welcome to Release 12.2 of the *Oracle E-Business Suite Extending Enterprise Command Centers.*

> **Note:** Some of the screenshots used in this book were captured on an earlier version. Although the images have the appearance of the earlier version, the functionality they illustrate also applies to the current version.

See Related Information Sources on page xiv for more Oracle E-Business Suite product information.

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 customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Structure

Related Information Sources

This book is included in the Oracle E-Business Suite Documentation Library. If this guide refers you to other Oracle E-Business Suite documentation, use only the latest Release 12.2 versions of those guides.

Online Documentation

All Oracle E-Business Suite documentation is available online (HTML or PDF).

- **Online Help** - Online help patches (HTML) are available on My Oracle Support.

- **Oracle E-Business Suite Documentation Library** - This library, which is included in the Oracle E-Business Suite software distribution, provides PDF documentation as of the time of each release.

- **Oracle E-Business Suite Documentation Web Library** - This library, available on the Oracle Help Center (https://docs.oracle.com/cd/E26401_01/index.htm), provides the latest updates to Oracle E-Business Suite Release 12.2 documentation. Most documents are available in PDF and HTML formats.

- **Release Notes** - For information about changes in this release, including new features, known issues, and other details, see the release notes for the relevant product, available on My Oracle Support.

- **Oracle Electronic Technical Reference Manual** - The Oracle Electronic Technical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for each Oracle E-Business Suite product. This information helps you convert data from your existing applications and integrate Oracle E-Business Suite data with non-Oracle applications, and write custom reports for Oracle E-Business Suite products. The Oracle eTRM is available as an application in Oracle E-Business Suite.

Related Guides

You should have the following related books on hand. Depending on the requirements of your particular installation, you may also need additional manuals or guides.

Oracle Application Framework Developer's Guide

This guide contains the coding standards followed by Oracle E-Business Suite Development to create applications with Oracle Application Framework. This guide is available in PDF format on My Oracle Support, and as online documentation in JDeveloper 10g with Oracle Application Extension.

Oracle Application Framework Personalization Guide

This guide covers the design-time and runtime aspects of personalizing applications built with Oracle Application Framework.

Oracle E-Business Suite: Administering Enterprise Command Centers

This book describes various tools and features of Oracle Enterprise Command Center Framework that can be used to manage and monitor Enterprise Command Centers.

Oracle E-Business Suite Concepts

This book is intended for all those planning to deploy Oracle E-Business Suite Release 12.2, or contemplating significant changes to a configuration. After describing the Oracle E-Business Suite architecture and technology stack, it focuses on strategic topics, giving a broad outline of the actions needed to achieve a particular goal, plus any installation and configuration choices that are available.

Oracle E-Business Suite Electronic Technical Reference Manual User's Guide

This guide describes how to set up and navigate Oracle E-Business Suite Electronic Technical Reference Manual (eTRM) user interface in Oracle E-Business Suite. It also explains how to browse and search the Oracle eTRM repository to locate desired FND and database metadata and objects, and how to view object details, reports, and diagrams.

Oracle E-Business Suite Flexfields Guide

This guide provides flexfields planning, setup, and reference information for the Oracle E-Business Suite implementation team, as well as for users responsible for the ongoing maintenance of Oracle E-Business Suite product data. This guide also provides information on creating custom reports on flexfields data.

Oracle E-Business Suite Security Guide

This guide contains information on a comprehensive range of security-related topics, including access control, user management, function security, data security, secure configuration, and auditing. It also describes how Oracle E-Business Suite can be integrated into a single sign-on environment.

Oracle E-Business Suite User's Guide

This guide explains how to navigate products, enter and query data, and run concurrent requests by means of the user interfaces (UI) of Oracle E-Business Suite. It includes basic information on setting preferences and customizing the UI. An

introduction to Oracle Enterprise Command Centers is also included. Lastly, this guide describes accessibility features and keyboard shortcuts for Oracle E-Business Suite.

Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the Oracle E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

Do Not Use Database Tools to Modify Oracle E-Business Suite Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle E-Business Suite data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle E-Business Suite data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle E-Business Suite tables are interrelated, any change you make using an Oracle E-Business Suite form can update many tables at once. But when you modify Oracle E-Business Suite data using anything other than Oracle E-Business Suite, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle E-Business Suite.

When you use Oracle E-Business Suite to modify your data, Oracle E-Business Suite automatically checks that your changes are valid. Oracle E-Business Suite also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

1

Introduction

Introduction to Oracle Enterprise Command Center Framework

Oracle Enterprise Command Center Framework provides information discovery, visualization, and exploration capabilities embedded within the Oracle E-Business Suite user interface. Oracle Enterprise Command Center Framework enables the creation of business dashboards in different functional areas such as Supply Chain, Financials, Manufacturing, Procurement, and so on.

In an Enterprise Command Center dashboard, users navigate transactional information using interactive visual components and guided discovery that allow exploratory data analysis. The ability to drill down to detailed insights into the data allows the dashboards to be action driven.

Each Oracle Enterprise Command Center has a set of features that allow users to answer essential questions, uncover business insights, and drill down to take the relevant action. It helps users quickly and intuitively analyze data so they can make confident decisions driven by data.

Oracle E-Business Suite delivers out-of-the-box dashboards across many product families and functional areas. The dashboards are rich in functionality and the nature of the technology allows for a highly interactive user experience that can cater to various use cases and scenarios. At the same time, the dashboards are extensible, allowing users to extend the use of the feature set and address additional requirements specific to their implementation.

This guide highlights important aspects of Oracle Enterprise Command Center Framework architecture and the anatomy of an Oracle Enterprise Command Center dashboard. It details the possible extension scenarios that can be achieved. It recommends best practices to be followed to preserve extensions after patching.

2

Oracle Enterprise Command Center Framework Architecture

Overview of Oracle Enterprise Command Center Framework Architecture

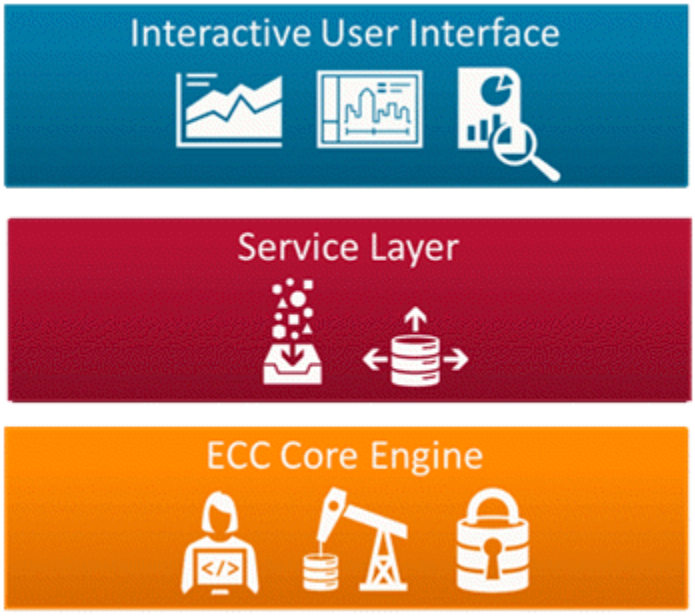
Oracle Enterprise Command Center Framework is a middleware technology made up of an interactive JavaScript front end and a Java back end engine. The architecture has three main components:

1. Interactive User Interface layer. It is built on top of the open source technology Oracle JavaScript Extensions Toolkit, JET. The Oracle Enterprise Command Center Framework development framework allows for code-free creation of dashboards using drag-and-drop layout design, declarative configuration of all components and automatic productive interactions across different sections of a dashboard.

2. Service Layer. Oracle Enterprise Command Center Framework provides services that allow the client to define a data set, ingest data into it, and control advanced behaviors using metadata. Additional services exposed to the dashboards allow advanced query execution against the data and return aggregations, rollups, associated data elements, and so on.

3. Oracle Enterprise Command Center Framework Core Engine. This is the core component of Oracle Enterprise Command Center Framework where the data ingested from the source system, that is, Oracle E-Business Suite, is stored. During data storage, analysis is performed to allow fast and advanced data access supporting several patterns of user queries. Advanced capabilities for language processing can optionally be employed to support dictionary based data parsing and retrieval.

Oracle Enterprise Command Center Framework Architecture



To create an Enterprise Command Center dashboard, the following steps are needed:

1. Define an application and relevant data sets with all metadata required to render pages and control every attribute behavior on the user interface. Advanced relationships, calculated attributes and dynamic UI interaction can further be specified.

2. Data ingested from the source system, that is, Oracle E-Business Suite, into the Oracle Enterprise Command Center Framework data set defined above.

3. Pages that expose ingested data and takes advantage of interactive components available in the Oracle Enterprise Command Center Framework component palette to build business dashboards.

3

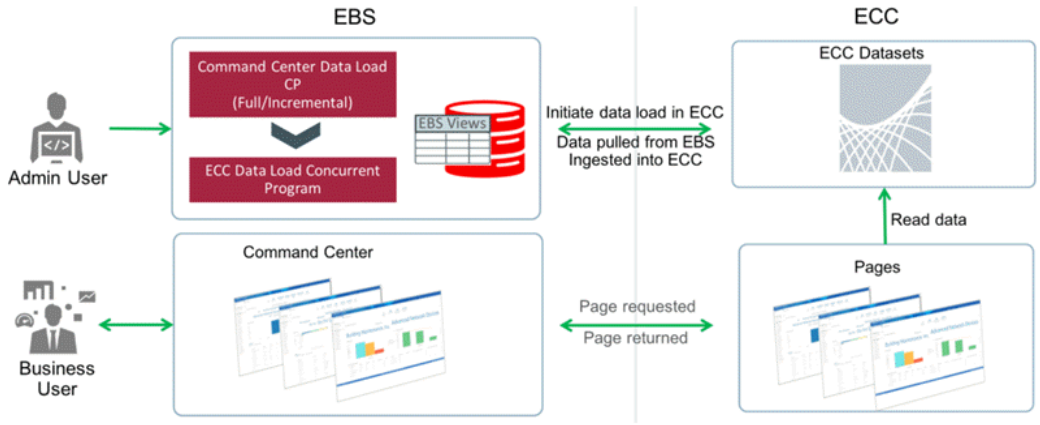
Data Load Process

Overview of a Data Load Process

Data is the core foundation for any dashboard. The data is ingested into the Oracle Enterprise Command Center Framework engine from the source system using rules and shapes defined by the source system. The data is modeled in a way to fit various use cases addressed by the dashboard. The source of truth of the data is the Oracle E-Business Suite system. Oracle Enterprise Command Center Framework architecture relies on the Oracle E-Business Suite database to run the data load process from Oracle E-Business Suite into Oracle Enterprise Command Center Framework. It does not introduce new complex tools or technologies but rather relies on the power and flexibility of SQL and PL/SQL to prepare and retrieve the data in the shape required to power the dashboards.

The following diagram shows the high-level architecture of a data load program. An administrator can use concurrent programs to trigger data load execution in Oracle Enterprise Command Center Framework to pull Oracle E-Business Suite data into ECC data sets. This ingestion of data lets can be done in a full data load or incremental data loads. A business user benefits from the data ingested into Oracle Enterprise Command Center Framework through the Enterprise Command Center. The Enterprise Command Center sends requests for pages that read data from the ECC data sets, and these pages are then returned to the Enterprise Command Center.

High-Level Architecture of Data Load Programs



Oracle Enterprise Command Center Framework supports online data load for minimizing the impact of downtime. Users can still use the dashboard while data load runs in the background. This feature works with the dual file system: an active/query data set for serving the queries, another data set for the extract, transform, and load (ETL) process. Once the full load process is completed, the query data set is deleted and ETL data set replaces the active data set. For details on enabling online full load, refer to Enabling Online Full Load, page F-3.

Oracle Enterprise Command Center Framework supports sequential full load. This feature allows for a multi-language full load to be executed for one particular language per execution rather that for all configured languages at once. This approach provides better control on overhead on the database side and can result in better overall performance and optimal ETL query executions. For details on enabling sequential full load, refer to Enabling Sequential Full Load, page F-4.

Data Load Process Definitions

Full Load

When you first install Oracle Enterprise Command Center Framework and import an application for a particular business area, one of the first steps you do is run full data load. The process creates the relevant data set(s) inside the Oracle Enterprise Command Center Framework engine and starts the ingest process from the Oracle E-Business Suite database into Oracle Enterprise Command Center Framework. Full load also provides an opportunity to recreate the data set and repopulate it with fresh data if the data set has already been created.

A full load for a business dashboard is triggered by a concurrent job owned by the same business area. The program triggers the associated data set load rules defined in Oracle Enterprise Command Center Framework. Every data set can have more than one rule to

load data with each of those containing one or more queries against the Oracle E-Business Suite database. Oracle Enterprise Command Center Framework connects to Oracle E-Business Suite using the EBS SDK to allow the data load code to run any data preparation, cleansing and wrangling logic before data is ingested into Oracle Enterprise Command Center Framework.

Incremental Load

Incremental data load follows the same logic and procedure as that for data load with two main differences. First, incremental load expects that the data set is already created and its metadata be already captured and will fail if these conditions are not met. Second, incremental data load will pick up only the modified data from Oracle E-Business Suite as of the time of the last successful full or incremental load.

File Upload

Oracle Enterprise Command Center Framework allows ingesting data into the data set using a CSV file. File upload provides the flexibility to use a file with any custom text separator and is specific to the language of the data in the file. File upload can be used to completely refresh the data or just extend existing data in data set. For more information, refer to Ingesting Data, *Oracle E-Business Suite: Administering Enterprise Command Centers*.

Query Upload

Oracle Enterprise Command Center Framework allows ingesting data into the data set using an SQL query on any connected database. Query Upload can also completely refresh the data or extend existing data in data set. For more information, refer to Ingesting Data, *Oracle E-Business Suite: Administering Enterprise Command Centers*.

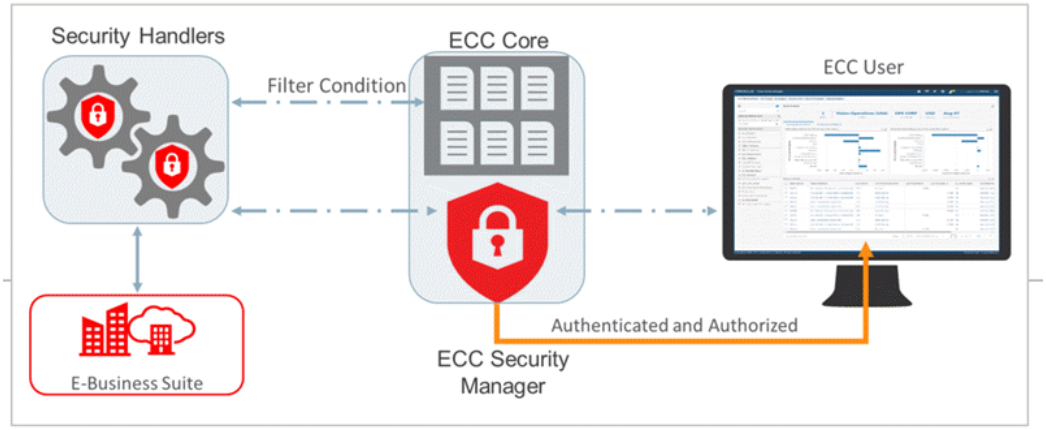
4

Security

Overview of Security in Oracle Enterprise Command Center Framework

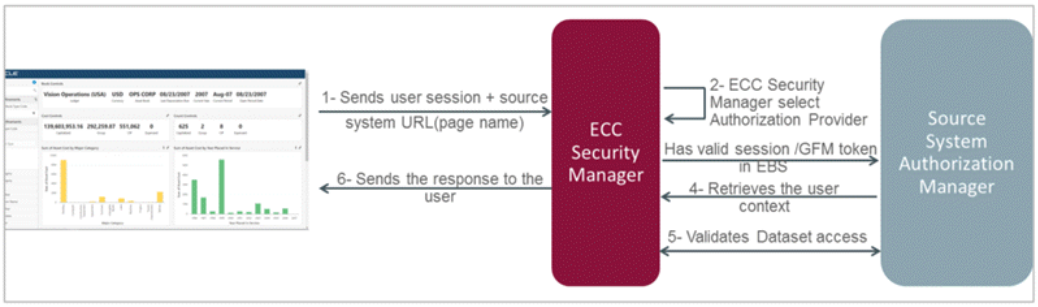
Oracle Enterprise Command Center Framework relies on the security model provided by Oracle E-Business Suite to secure access to dashboards and the underlying data. It inherits user access privileges from Oracle E-Business Suite and relies on the user being authenticated by a Security Manager and Oracle E-Business Suite before gaining access to the embedded Oracle Enterprise Command Center Framework dashboards. Filters are also applied to control access to dashboards and data sets.

Overview of Security Model for Oracle Enterprise Command Center Framework and Oracle E-Business Suite



Oracle Enterprise Command Center Framework Authentication

The figure below explains the process that Oracle Enterprise Command Center Framework employs to authenticate and authorize access to the embedded dashboard.



When a business user accesses an embedded dashboard, the following procedure is followed:

1. Oracle Enterprise Command Center Framework sends information on the user session and the source system URL (page name).
2. The ECC Security Manager selects an Authorization Provider.
3. The ECC Security Manager validates the existence of a valid and authenticated Oracle E-Business Suite session before any additional checks are performed. It sends a valid session token for Oracle E-Business Suite to the Source System Authorization Manager.
4. The ECC Security Manager then retrieves the user context.
5. Authorization controls are applied next to allow granular control over which dashboards are exposed to the user.

   The ECC Security Manager also validates access to data sets. Authorization is verified through matching the Oracle Enterprise Command Center Framework data set privilege name with the FND form function.
6. A response is then sent to the user.

Page Level Security

Pages are secured in Oracle Enterprise Command Center Framework using FND_FORM_FUNCTION defined in Oracle E-Business Suite.

A shipped Oracle E-Business Suite role (utilizing Role Based Access Control, or RBAC) is used to assign Oracle Enterprise Command Center Framework dashboard access to responsibilities.

Oracle Enterprise Command Center Framework ships an ECC Developer responsibility that allows access to the Oracle Enterprise Command Center Framework home page and supports different developer and administrator capabilities.

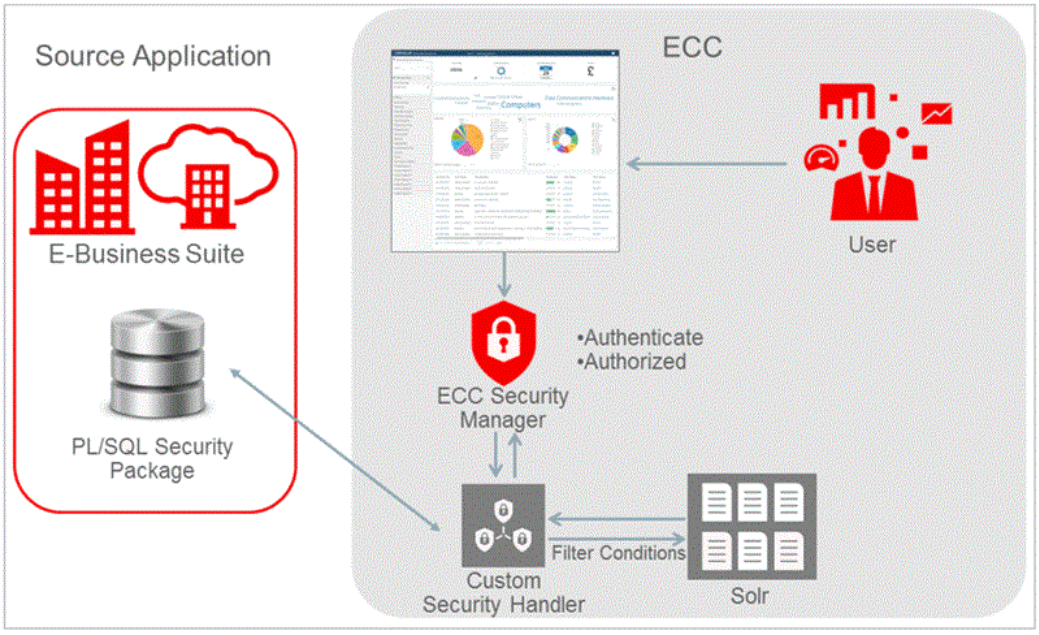
A dashboard page, when accessed by a user who has the ECC Developer responsibility, will allow edits to page layout and component configurations.

Data Level Security

To access data within an Oracle Enterprise Command Center Framework data set, the user must have at least one of the data set privileges (these are controlled and defined as an Oracle E-Business Suite form function). Once access to the data set is verified by the Enterprise Command Center Framework Security Manager, a security handler is applied for any subsequent query against that data set to ensure data access is restricted to the subset the user is allowed to see. This is the next level of security that enforces data access restrictions on data displayed in the dashboard based on setups available in Oracle E-Business Suite that may be applicable to different business dashboards (for example operating unit access, inventory org access, asset book access, and so on).

As illustrated in the following diagram, Oracle Enterprise Command Center allows configuration of a custom security handler that is applied on top of the existing security handler. A separate package can be configured as the custom security handler, utilizing a PL/SQL security package in Oracle E-Business Suite. This custom security handler can also use Apache Solr as a search API.

Data Security



5

Metadata

Overview of Metadata in Oracle Enterprise Command Center Framework

The behavior of an Oracle Enterprise Command Center Framework dashboard, its underlying components, and the data elements included within is controlled through metadata. Metadata is specified in different parts of the developer/admin UI. Some of the metadata is defined at design time and referred to plainly as metadata. Additional metadata can be defined dynamically at runtime to control behaviors not known or cannot be specified at design time. This type is referred to as dynamic metadata.

Data Set

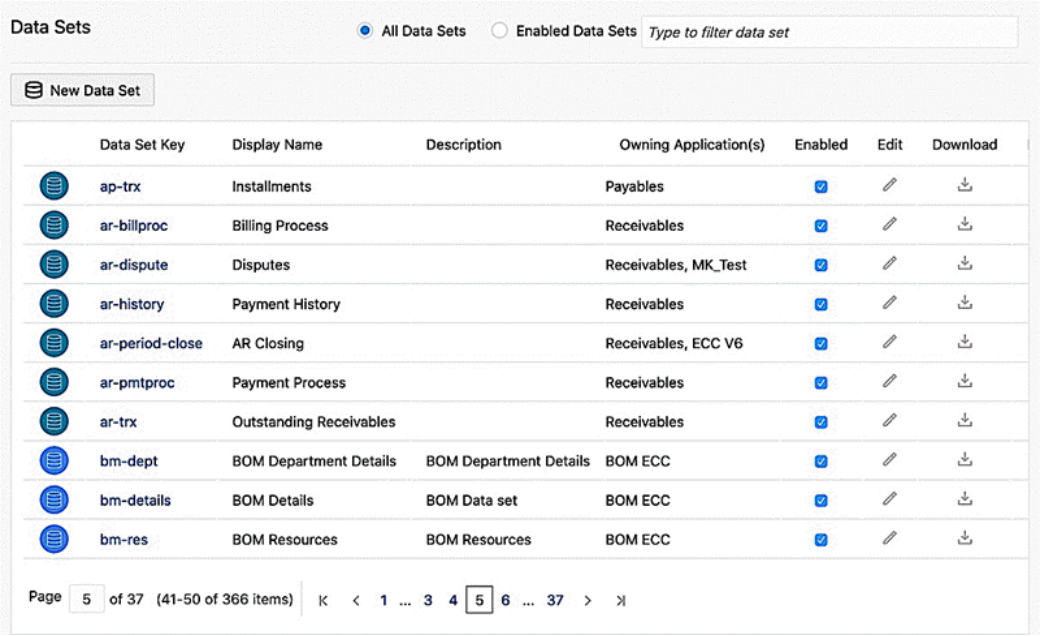
A data set is both a logical and a physical grouping of attributes to support business dashboard operations and use cases. From a logical perspective, it is designed to support several use cases that are accessed through one or more Oracle Enterprise Command Center Framework dashboards. The design typically caters to open-ended interaction with the underlying data.

At the physical level, the data set stores one or more records with a uniquely identifying key that represents a particular level of detail of the entity stored in the enterprise system.

A data set declares the data load rules that it supports and how they are used to populate data into the data set. Each data set should be assigned to an application as an owned data set but can be referenced by other applications. The owning application is responsible for populating data into the data set.

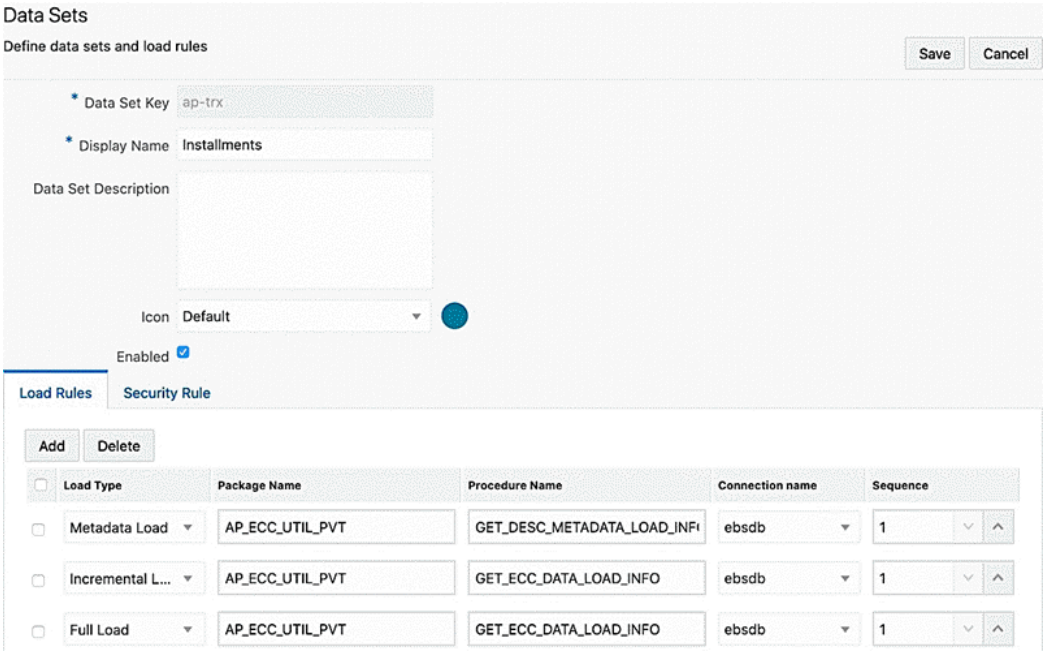
A data set's contents can be downloaded into a CSV file by clicking the Download icon.

Data Sets Page



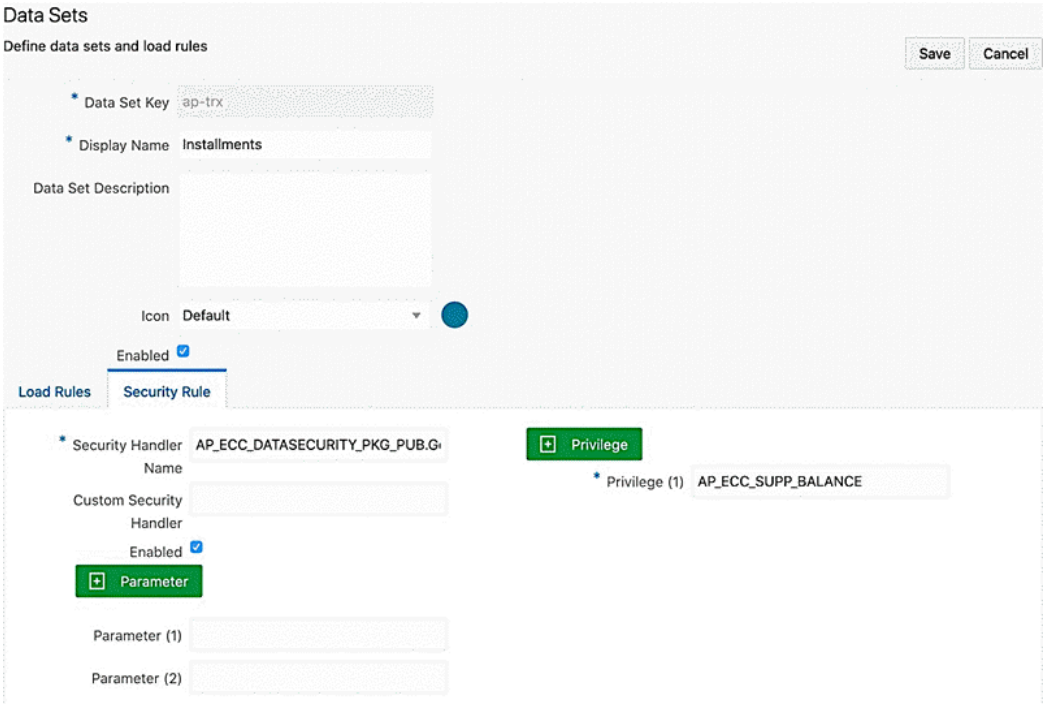
Beginning with Oracle Enterprise Command Center Framework V6, each data set can be represented using an appropriate icon. Each data set has Load rules (Data Load process).

Data Set Load Rules



Each Data set also has a security rule (Security) where a security handler is defined along with privileges. For more information, see Overview of Security in Oracle Enterprise Command Center Framework, page 4-1.

Data Set Security Rules



Contents of a data set can be exported with attribute keys and attribute display names as headers. For more information refer to Highlights of an Enterprise Command Center, *Oracle E-Business Suite User's Guide*.

> **Note:** A data set cannot be deleted if it is used to configure one or more components.

Metadata

Each attribute stored in the data set is controlled by metadata properties that specify its behavior on the user interface. Additional value-add features such as calculations, bucketing, and precedence rules can also be specified.

Attribute Metadata

Attribute metadata define the attribute's characteristics, including:

- Name and type.
- Display name.
- Configuration parameters. For example, whether an attribute is searchable.

- Navigability settings. For example, whether to show record counts for available refinements, whether to enable multi-select and how to sort refinements.

Attribute Level Properties

Metadata properties on the attribute level are listed in the following table:

Metadata Attribute-Level Properties

Property	Default Value	Description
Attribute Key	N/A	The key of the attribute as defined in data load queries. This attribute is mandatory and unique per data set.
Source Data Type	N/A	The attribute data type as defined in data load queries.
Profile	N/A	Every attribute has a profile associated with it that includes further properties controls the attribute behavior. Two new profile types are introduced in V6: • dateTime: This allows configuration of attributes to display date along with timestamp. • dateTimes: This profile type supports multi assignment of dateTime attributes.
Display Name	N/A	The name of the attribute in an easy-to-understand format. The user can modify the display name of an attribute. No restrictions on these changes exist.
Custom Display Name	N/A	Used to override the shipped attribute display name.

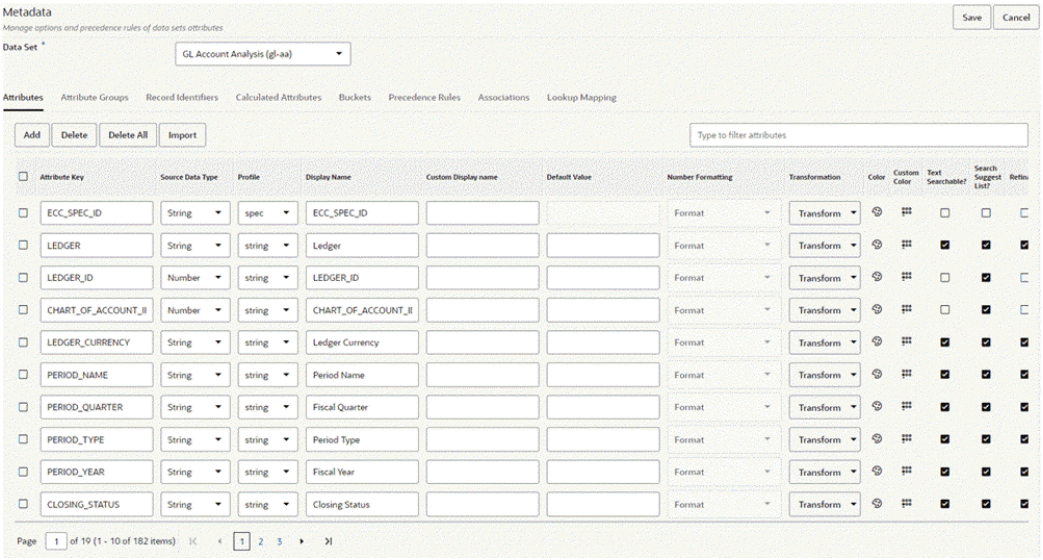
Property	Default Value	Description
Default Value	N/A	Beginning with V6, this flag is used to replace the null values present in the attribute data. Default Value is a free text and is applicable for profiles: String, Date, DateTime, Int, Float. Default Values behave like any other value in runtime. Replacement of null values with default values happens on the fly during a full load. Once the null value is replaced, the default value behaves as any other value and is displayed in all UI components, thereby improving data quality and allowing a business user to filter using this value.
Number Formatting	N/A	Beginning with V11, this option provides additional formatting options for attributes of type Int/Ints, Double/Doubles, and Long/Longs. Formatting is also applicable to calculated attributes. Three types of formatting are supported: General, Formatted Number, and Accounting. Refer to tables below for more information on number formatting.

Property	Default Value	Description
Transformation	N/A	Beginning with ECC V8, this option enables a date subset for all date and date/time attributes. It supports different levels (Year, Quarter-Year, Month-Year, Quarter, Month). Attributes are created after enabling the date subset transformation option is enabled and saved. A full load is required after enabling the date subset transformation option. The new subset attributes (Year, Quarter-Year, and Month-Year, Quarter, Month) are described below. • For the Attribute Key <ATTRIBUTE_KEY>YEAR, the Display Name is Attribute Display Name (Year). • For the Attribute Key <ATTRIBUTE_KEY>QUARTER, the Display Name is Attribute Display Name (Quarter). • For the Attribute Key <ATTRIBUTE_KEY>QUARTER_YEAR, the Display Name is Attribute Display Name (Quarter-Year). • For the Attribute Key <ATTRIBUTE_KEY>MONTH, the Display Name is Attribute Display Name (Month). • For the Attribute Key <ATTRIBUTE_KEY>MONTH_YEAR, the Display Name is Attribute Display Name (Month-Year). Date subset attributes are displayed in the attribute metadata page as row expander read-only records, and the "Attribute Key", "Source Data Type", and "Profile" options are disabled. Date subset attributes can be used in the component configuration.

Property	Default Value	Description
Color	N/A	Beginning with V6, this is used to pin colors on specific attribute values. The pinned colors remain intact on these values irrespective of where the chart is configured. In V10, a new color palette is introduced for color pinning in the Results Grid and Results Table components, in addition to a new palette for dataset color, summary bar flag color, and metric colors in components.
Custom Color	N/A	This property is used to override the shipped color pinning. Designers can take advantage of this property to pin context-specific colors.
Text Searchable	Y	If set to Y, then the attribute is enabled for the record search. If set to N, the attribute does not support record search.
Search Suggest List	Y	Beginning with V6, if this attribute is set to Y, then the attribute is included in value search suggestions in runtime. The flag is set to Y by default for String type attributes.
Refineable?	Y	Specifies whether attributes are enabled to be used as refinements. If set to Y, the attribute is refineable. If set to N, the attribute is not refineable.
Translatable?	N	Specifies whether the attribute is translatable or not.
Refinement Behavior	multi-select-or	Configures the multi-select feature for an attribute, and can be overridden with the following values (single, multi-or, multi-and).

Property	Default Value	Description
Refinement Order	lexical	The order in which to display refinements in the navigation menu. The allowed values are: • lexical - sorts refinements alphabetically or by number. • record-count - sorts refinements in descending order, by the number of records available for each refinement.

Metadata Attributes Tab



Number Formatting

As described above, beginning with V11, designers can specify the formatting of numerical attributes so that these attributes are displayed uniformly in a dashboard.

Three types of number formatting are supported: General, Formatted Number, and Accounting.

For a numeric attribute, a designer specifies a Profile Type of `Int/Ints`, `Long/Longs`, or `Double/Doubles`. and then chooses the type of number formatting. These options are described in the tables below.

Number Formatting Types for Profile Type `Int/Ints`

Formatting Type	Formatting Logic	Example
No Formatting	Comma (,) as the thousands separator	`10,000` `-10,000`
General	Absence of the thousands separator	`10000` `-10000`
Formatted Number	Comma (,) as the thousands separator	`10,000` `-10,000`
Accounting	Comma (,) as the thousands separator Negative numbers within parentheses ()	`10,000` `(10,000)`

Number Formatting Types for Profile Type `Long/Longs`

Formatting Type	Formatting Logic	Example
No Formatting	Comma (,) as the thousands separator	`100,000` `-100,000`
General	Absence of the thousands separator	`100000` `-100000`
Formatted Number	Comma (,) as the thousands separator	`100,000` `-100,000`
Accounting	Comma (,) as the thousands separator Negative numbers within parentheses ()	`100,000` `(100,000)`

Number Formatting Types for Profile Type `Double/Doubles`

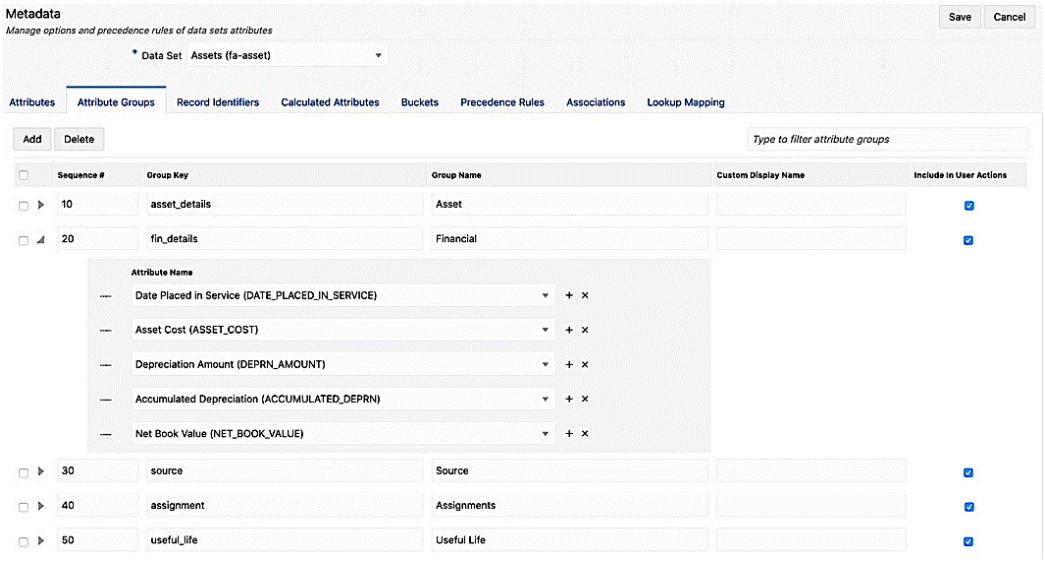
Formatting Type	Formatting Logic	Example
No Formatting	Comma (,) as the thousands separator	`100,000.00` `-100,000.1`
General	Absence of the thousands separator Non-uniform representation of decimals	`100000` `-100000.1`
Formatted Number	Comma (,) as the thousands separator Uniform decimal representation to two (2) places	`100,000.00` `-100,000.10`
Accounting	Comma (,) as the thousands separator Negative numbers within parentheses ()	`100,000.00` `(100,000.10)`

Attribute Groups

Attribute Groups are logical groupings of attributes for display purposes based on functional/business meaning. They reduce clutter on Available Refinements and Results Table components.

Attribute Groups are defined at the data set level, where each data set can have zero or more attribute groups, and each group can have zero or more attributes. Groups can be designed at design time in the Administration UI or during runtime in metadata load phase.

Attribute Groups Tab



Record Identifiers

As a data set can be highly denormalized, a record identifier is an easy yet powerful feature that allows the display of data set records at different levels of granularity. It employs a similar concept to a grouping key in SQL queries.

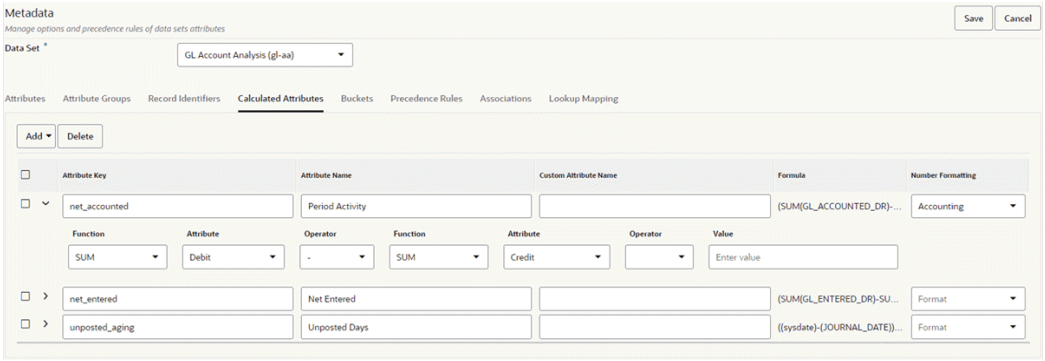
Calculated Attributes

You can define calculated attributes in metadata to allow for dynamic behavior and reduce the complexity of ingested data; there are two types of calculated attributes:

- Record-based Date function
  - Facilitates time-based bucketing
  - Uses `sysdate` in the date calculation
  - Calculation result unit can be specified as day, month, or year
  - Can be used in any component as a data condition at the component configuration level
- Group-based Aggregation function
  - Allows on the fly aggregations that cannot be computed during data load
  - Grouped by dimension selected in the component configuration

- Operate on a group of records
- Used as a metric (without a need for further aggregation) in visualization components

Calculated Attributes Tab



Beginning with V9, Oracle Enterprise Command Center Framework supports:

- Aggregation on the 'Date/Date Time' attribute
- Calculation using two different date-based calculated attributes
- Use of the plus sign (+) operator to create an offset on any date

Examples are:

- Recency of a customer, which is a difference between current time and the last time when the customer has used the product or when is the last time the customer, has raised a service request: `Recency = sysdate - max (SR created date)`
- Response time of a service request, which can be calculated using the difference between two date attributes: the day the service request was closed and the day the service request was filed: `Response Time = Closed Date - Created Date`

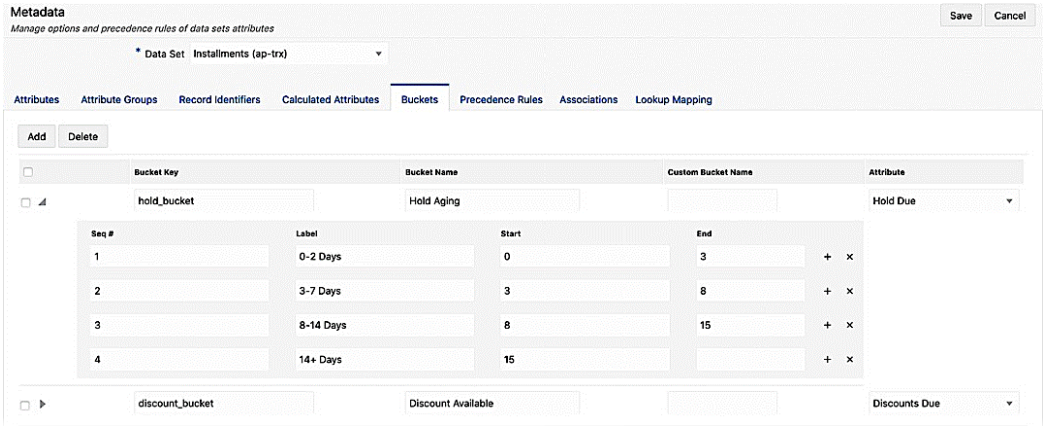
Buckets

Buckets allow binning of data records based on a particular metric falling within the specified range. The controlling metric can be a regular or calculated attribute available in the data set.

- Sequence numbering controls the display order.
- An inclusive range start and exclusive range end can be specified.
- An empty start/end means positive or negative infinity.

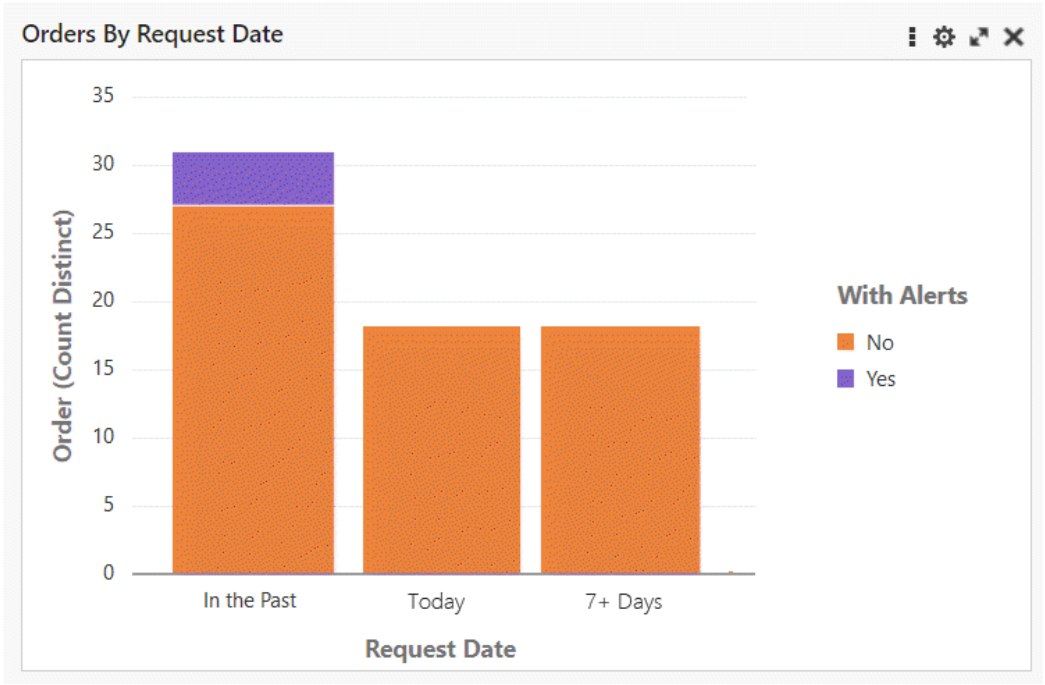
- Gaps and intersections are allowed.

Buckets Tab



A bucket effectively creates a dynamic dimension on the record which can then be used in a Chart for display and refinement purposes.

Chart with Bucket



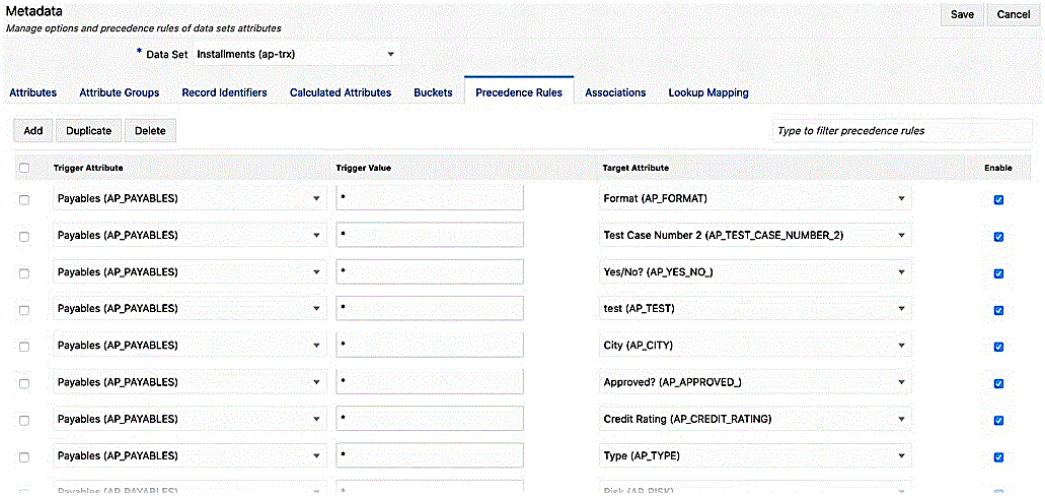
Precedence Rules

Precedence rules control dynamic disclosure of additional attributes in the Available Refinements component. They provide a way to display additional attributes only when

a certain data condition of a trigger attribute is met.

Precedence rules are defined in terms of a trigger attribute and a target attribute, where the trigger attribute reveals an additional target attribute to the user.

Precedence Rules

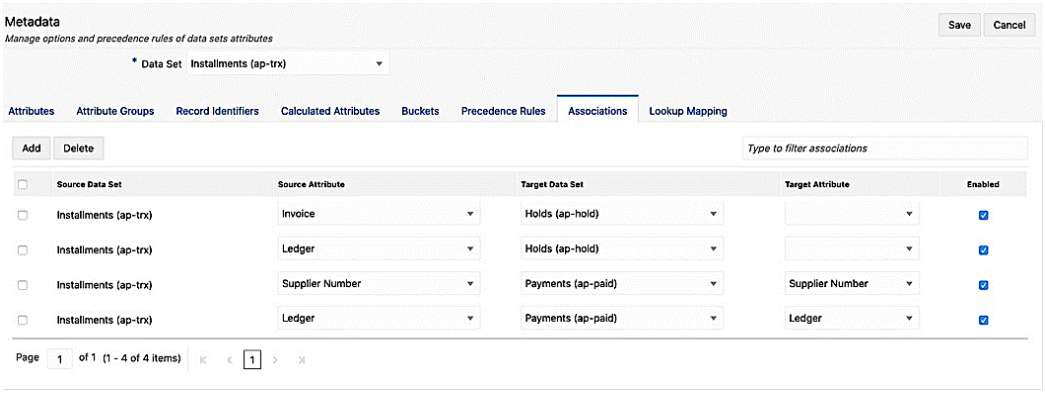


Associations

Associations allow explicit connection between two data sets by defining a link between a common attribute across the data sets. Oracle Enterprise Command Center Framework utilizes this information to enforce associative filtering of data in one destination data set based on refinement state in a source data set.

Additionally, Oracle Enterprise Command Center Framework establishes an implicit relationship between two different data sets that are both associated with a shared data set but not directly linked with each other. This advanced capability allows for advanced refinement state inheritance between data sets that are not directly associated with each other.

Associations



Lookup Mapping

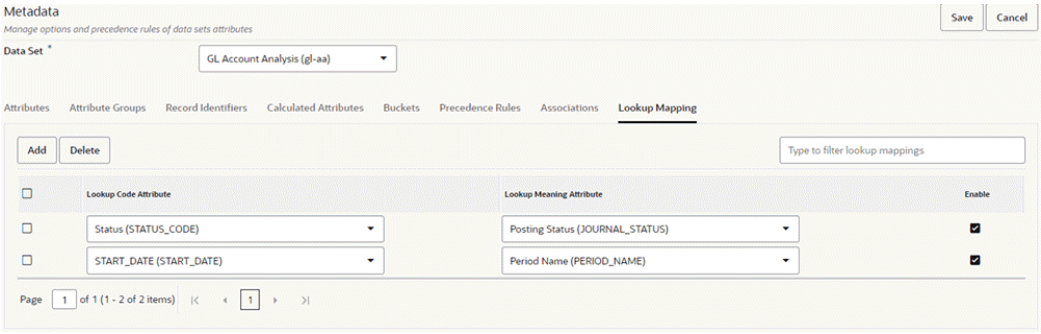
Lookup Mapping allows for the connection between two attributes by defining a link between a common attribute across the data sets. Oracle Enterprise Command Center Framework utilizes this information to enforce associative filtering of data in one destination data set based on the refinement state in a source data set.

This capability improves the readability in Refinements. The Refinements feature uses Lookup Mapping to display attribute descriptions in tooltips when the attribute code and attribute descriptions have a mapping.

Color Pinning also utilizes Lookup Mapping to display the same pinned colors across different user languages. This can be achieved by configuring color pinning on non-translatable attribute code and defining mapping between attribute code and a translatable attribute description.

From V11 onwards, the scope of lookup mapping is extended to support alphanumeric sorting in Oracle Enterprise Command Center Framework. Alphanumeric sorting uses the link between the code and meaning attribute to apply sorting on "alphanumeric values" across a dashboard.

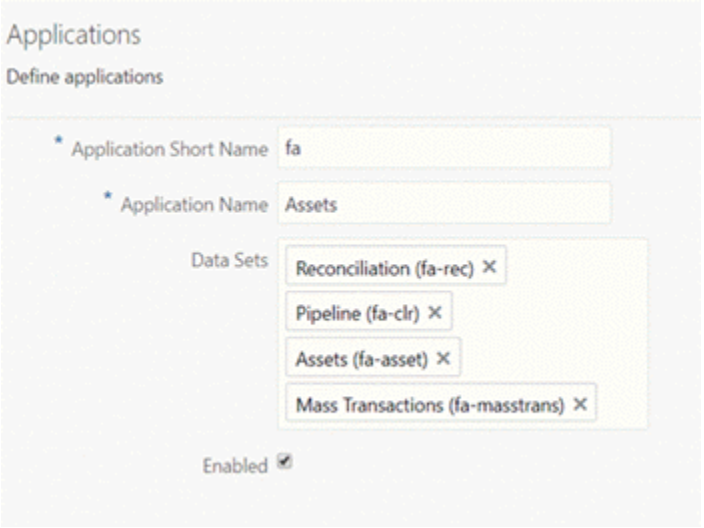
Lookup Mapping Tab



Application

A Command Center is made up of several dashboards exposing different aspects of a functional area. An Oracle Enterprise Command Center Framework application is the logical grouping of the artifacts making up a command center, such as dashboards and the relevant data sets.

Application Definition



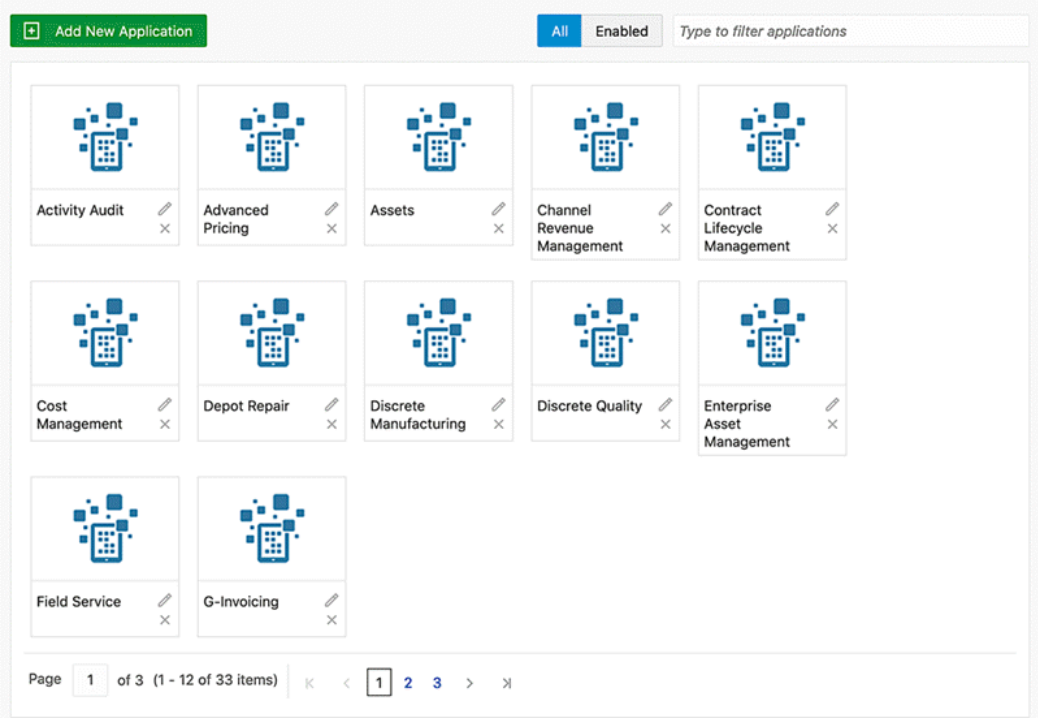
An application encapsulates all elements needed to power the dashboard. It references data sets which in turn control how data is populated through load rules and behavior of every attributes on the UI through metadata. An application can reference data sets owned by other applications.

Page

An application page is home for all the visualization components that are designed to

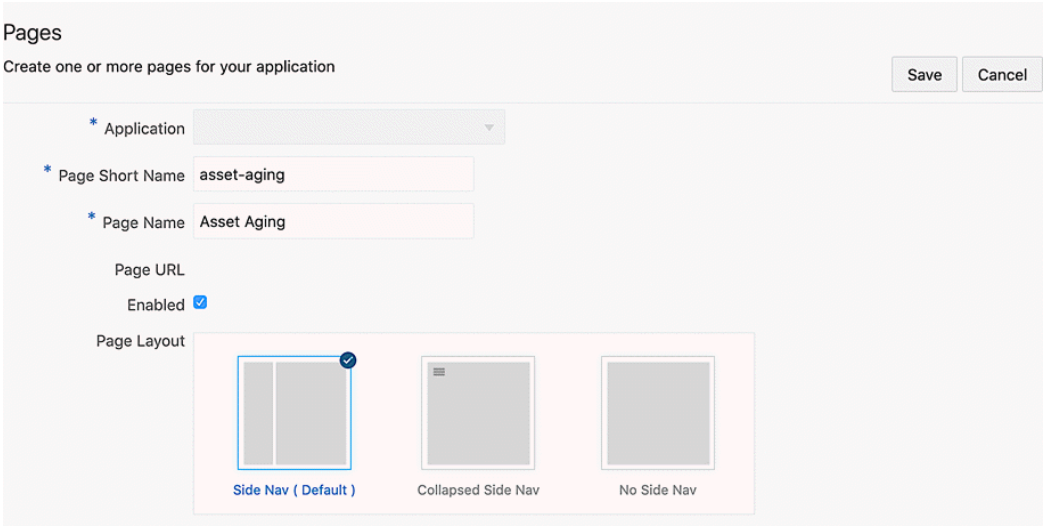
perform a specific type of function: Filtering the data displayed on the page, displaying visual representations of data, displaying lists of records or record attributes, or highlighting specific values.

Application Pages



Pages are grouped under application as per the business requirement. Page definitions can be configured with the essential details such as: page name, short name, and page layout.

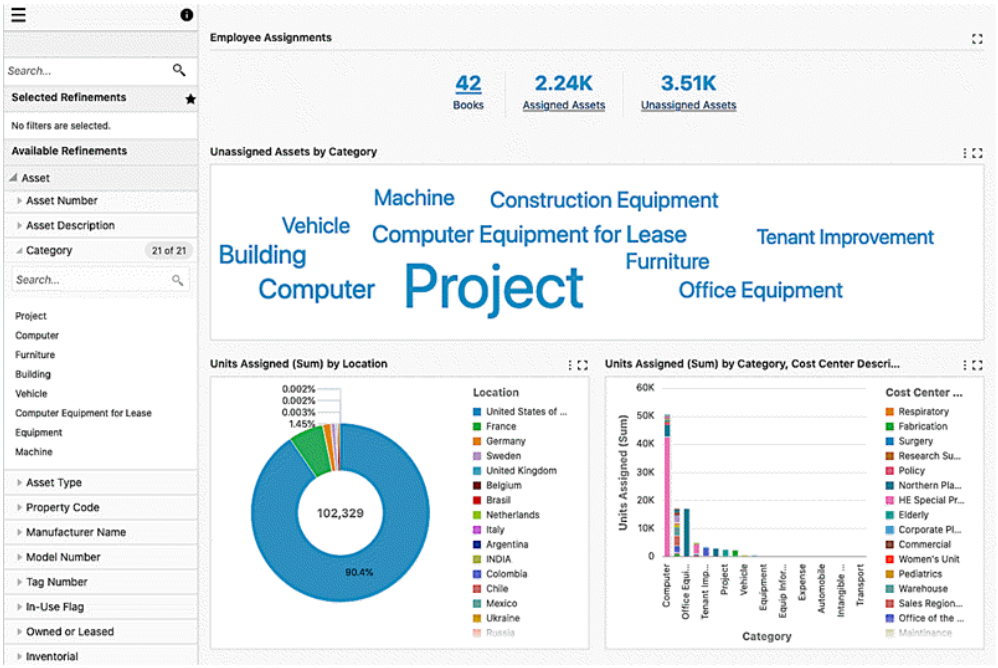
Example of a Page Layout



Three different types of layout configuration are available for the pages:

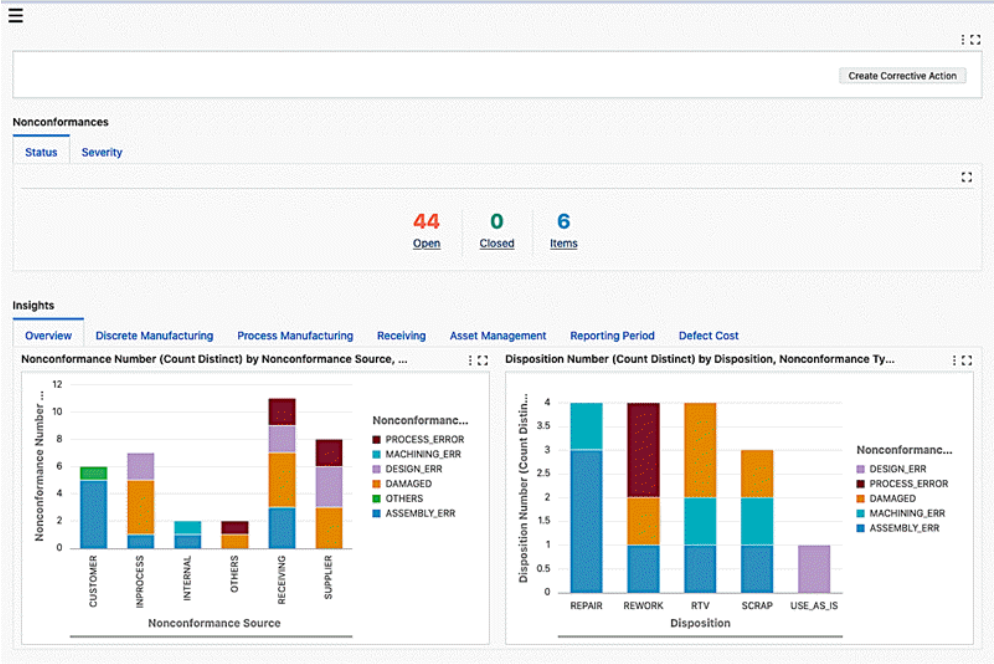
- Layout with side navigation

Example of Page with Side Navigation



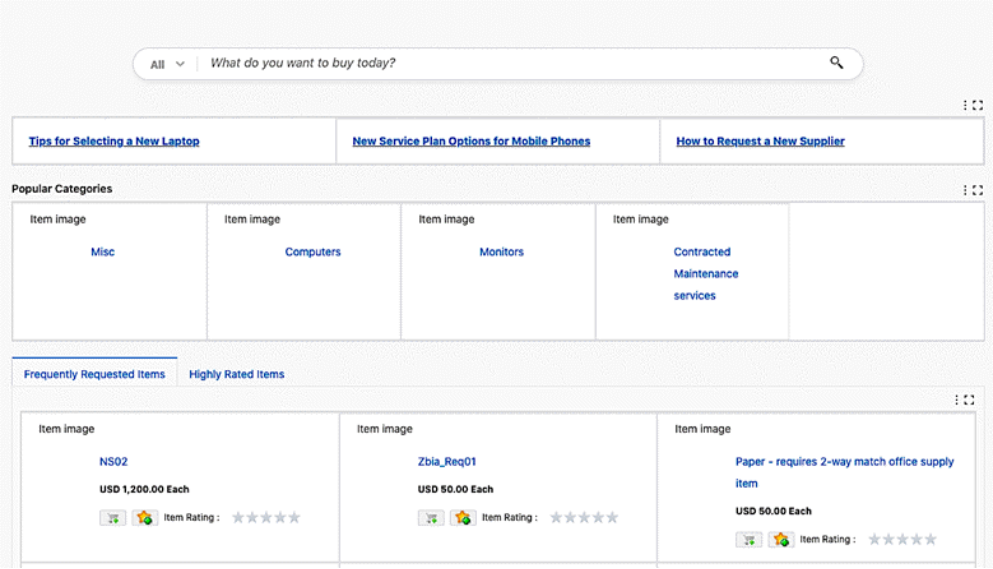
- Layout with collapsed side navigation

Example of Page with Collapsed Side Navigation



- Layout with no side navigation.

Example of Page with No Side Navigation



Side navigation is enabled in the default layout. When the navigation panel is hidden, the navigation icon will not take vertical space from the page and takes shape of a hamburger icon.

When the side navigation is collapsed, all the selected refinements are displayed in a funnel icon. The same is applicable for the layouts: collapsed side navigation and no side navigation by default. When a refinement is applied on the page, the selected refinement funnel icon changes to indicate the filters applied.

Overview of Components

An Oracle Enterprise Command Center Framework dashboard is a collection of UI visualization components.

Note that in V10, user experience has been re-designed to be aligned with new Oracle design standards. These updates include a new color palette, new interaction elements, new font types and font sizes, improvements in individual component designs, and overall improvements to dashboard level and administration level look and feel.

Oracle Enterprise Command Center Framework UI components are grouped into four main groups:

Component Groups

Group	Components
Navigation	Search Box
	Selected Refinements (breadcrumbs)
	Available Refinements
Visualization	Summarization Bar
	Chart
	Tag Cloud
	Aggregated Table
	Diagram
	Aggregated Grid
Detailed Insight	Results Table
	Grid
Layout	Tabbed Component Container

All Oracle Enterprise Command Center Framework components come with a unified configuration model that enhances the configuration experience. Also, many components share a subset of user-facing options.

6

Dynamic Metadata

Dynamic Attributes

As discussed in the previous chapter, in some cases you may wish to define metadata at runtime. This allows for behaviors that are dynamically controlled. This pattern fits well with user-defined attributes.

To define dynamic attributes, you should have a metadata load phase defined for the data set. Each time the user runs the metadata load, Oracle Enterprise Command Center Framework cleans up all dynamic attributes and inserts a fresh set of dynamically defined attributes. Standard defined attributes that ship with application definition are not affected by this refresh process.

You can define dynamic attributes in either one of two approaches:

- **Describe Option**: A query that is parsed by Oracle Enterprise Command Center Framework to define metadata of the attributes retrieved. No actual execution of the query happens.
- **Execute Option**: A query that is executed to define metadata of attributes.

Describe Option

This option builds the attributes list based on the SQL query select column list metadata. It retrieves all the columns in the select statement and reads the name, type, precision, and scale of each column and creates a corresponding attribute in the data set. If Oracle Enterprise Command Center Framework is configured to run in multi-language mode, then the attribute display name is captured as-is from the database, that output is then parsed, and the administrator is expected to provide translations per enabled language.

Attribute Mapping of the 'Describe' Function

Attribute Name	Default Value
ATTRIBUTE_KEY	The key of the attribute as defined in data load queries. This attribute is mandatory and unique per data set.
DISPLAY_NAME	Column Name, with an empty space replacing the underscore.
CUSTOM_DISPLAY_NAME	Empty string
PROFILE	Based on the type value. See the note below.
SEARCHABLE_FLAG	Attribute type String 'Y', else 'N'
REFINABLE	Attribute type String 'Y', else 'N'
SHOW_IN_GUIDED_DISCOVERY	'Y' in dynamic attributes, 'N' in standard attributes.
REFINEMENT_BEHAVIOUR	multi-select-or
REFINEMENT_ORDER	lexical
SHOW_RECORDS_COUNT	'N'
COLLAPSED	'Y'
TRANSLATABLE	'N'
LANGUAGE	Language of the logged-in user if the user runs it from the administration side; Oracle Enterprise Command Center Framework base language if the user runs it from the metadata load.
SOURCE_LANGUAGE	LANGUAGE

For "Profile" in the above table, the default value is based on the type value as follows:

```
String => String profile
Date => Date profile
```

```
Number => PrecisionSize > 0  => columnSize < 10 => float profile
                             => columnSize > 10 => double profile
       => PrecisionSize < 0  => columnSize < 10 => int profile
                             => columnSize < 10 => long profile
```

Execute Option

Runs the provided select statement then loops over the returned rows and creates a corresponding dynamic attribute per row. The service expects the below list of columns in the provided select statement.

The query should return all mandatory fields; otherwise, an exception with root cause details will be raised.

Attribute Mapping of the 'Execute' Function

Attribute Name	Default Value	Mandatory/ Optional	Options
NAME		Mandatory	
Type		Mandatory	
Precision		Mandatory	
Scale		Mandatory	
DISPLAY_NAME	Name, but replacing the underscore with empty space	Optional	
CUSTOM_DISPLAY_NAME	Empty String	Optional	
SHOW_IN_GUIDED_DISCOVERY	'Y' in dynamic attributes, 'N' in standard attributes	Optional	'Y', 'N'
REFINEMENT_BEHAVIOUR	multi-select-or	Optional	multi-select-or, multi-select-and, single-select
REFINEMENT_ORDER	lexical	Optional	Lexical, record count
SHOW_RECORDS_COUNT	'N'	Optional	'Y', 'N'

Attribute Name	Default Value	Mandatory/ Optional	Options
COLLAPSED	'Y'	Optional	'Y', 'N'
PROFILE	Based on the type value	Optional	One of the ATTR_PROFILE_KEY in the table ECC_DATASET_ATTR_PROFS
TRANSLATABLE	'N'	Optional	
LANGUAGE	Language of the logged-in user > if the user runs it from the admin side. Oracle Enterprise Command Center Framework Base language > if the user runs it from the metadata load.	Optional	One of Oracle Enterprise Command Center Framework languages added in ecc.languages in the ecc-config.properties file
SOURCE_LANGUAGE	Language of the logged-in user > the user runs it from the admin side. Oracle Enterprise Command Center Framework Base language > if the user runs it from the metadata load. Language > If the language existes and the supported language does not exist.	Optional	One of Oracle Enterprise Command Center Framework languages added in ecc.languages in the ecc-config.properties file
SEARCHABLE_FLAG	'N'	Optional	'Y', 'N'
REFINABLE	'N'	Optional	'Y', 'N'

Attribute Name	Default Value	Mandatory/ Optional	Options
SNIPPET_SIZE	0	Optional	Integer value
GROUP_KEY		Optional	

Dynamic Precedence Rules

Precedence rules control the disclosure of additional attributes based on a data condition being met. They can also be specified dynamically through an Execute Option only.

Specify a SQL statement that will be executed to return the definition of one or more precedence rules. The underlying service will insert a new rule if one does not exist for the same combination of source, target, and trigger values. Otherwise, it will update and enable an existing one.

In case of an empty value for any of the mandatory fields, the service throws an exception, stops the execution, and provides a clear message about the issue.

Dynamic Precedence Rules

Attribute Name	Mandatory/Optional
TRIGGER_INSTANCE_ATTRIBUTE	Mandatory
TRIGGER_ATTR_VALUE	Mandatory
TARGET_INSTANCE_ATTRIBUTE	Mandatory
ENABLED_FLAG	Optional

7

Anatomy of Oracle Enterprise Command Center Framework UI Components

Configuration Model

The Oracle Enterprise Command Center Framework configuration model uses a familiar structure that follows that of a SQL query. No coding or custom expressions are required to configure any Oracle Enterprise Command Center Framework component.

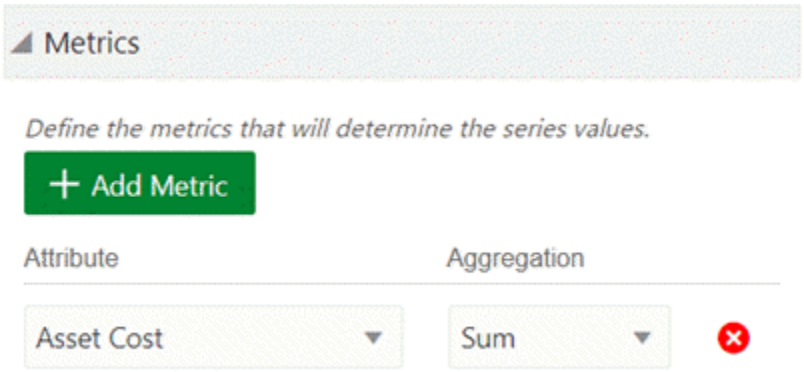
For the sake of explaining different configuration options of Oracle Enterprise Command Center Framework components, we will use a SQL query structure to introduce the relevant concept in the Oracle Enterprise Command Center Framework configuration template. Please note that Oracle Enterprise Command Center Framework does not execute a SQL query to display data in a component.

For example, given the SQL query:

```
SELECT
   SUM(ASSET_COST),
   MAJOR_CATEGORY
```

The metric can be added under "Metrics" with the Attribute of "Asset Cost" and Aggregation "Sum".

Example of Defining a Metric



If the SQL statement is:

```
FROM
  fa-asset
```

The Oracle Enterprise Command Center Framework Configuration would show `fa_asset` as the selected data set.

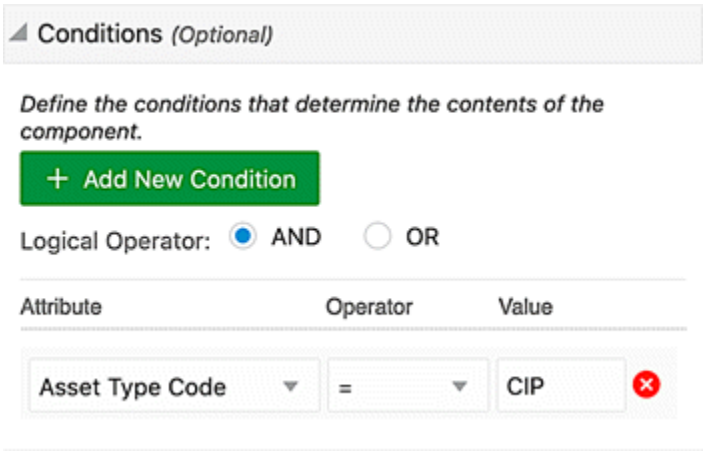
Example of Selecting a Data Set

If the SQL statement is:

```
WHERE
  ASSET_TYPE = 'CAPITALIZED'
```

The Oracle Enterprise Command Center Framework Configuration would have the condition with `Asset Type Code` for Attribute, = for Operator, and `CAPITALIZED` for Value.

Example of Setting the Condition

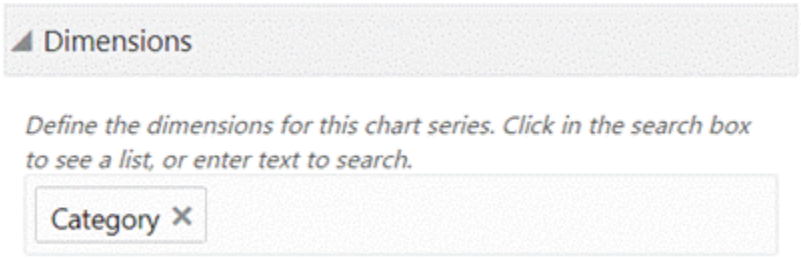


If the SQL statement is:

```
GROUP BY
  MAJOR_CATEGORY
```

The Oracle Enterprise Command Center Framework Configuration would have `Category` listed under Dimensions

Example of Defining the Dimensions

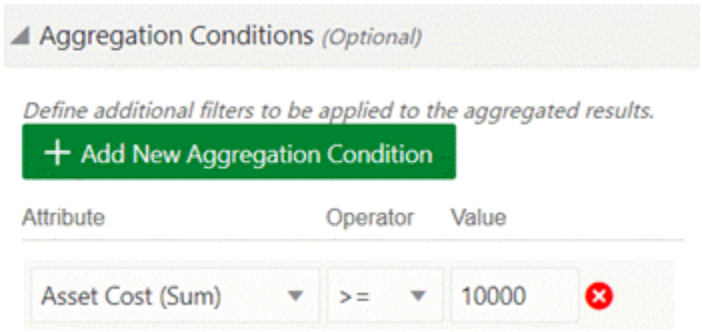


If the SQL statement is:

```
HAVING
  SUM(ASSET_COST) > 10000
```

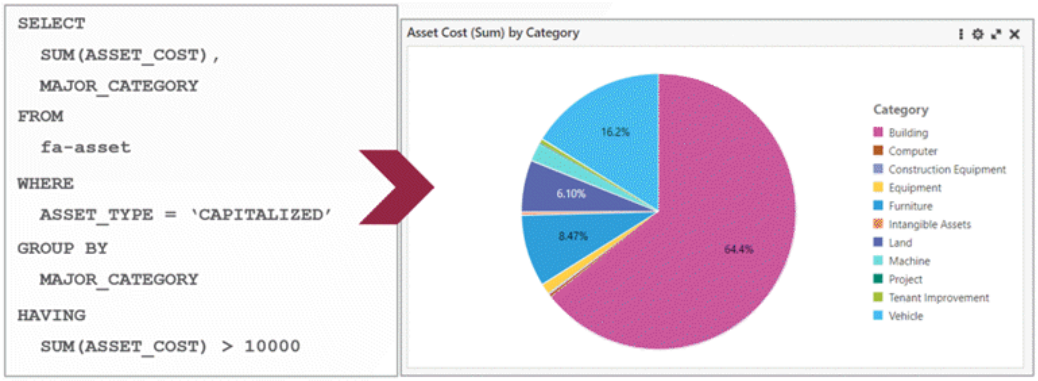
The Oracle Enterprise Command Center Framework Configuration would have `Asset Cost (SUM)` for Attribute, `>=` for Operator, and `10000` for Value under Aggregate Conditions.

Example of Defining the Aggregation Condition



Based on above example, the following chart will be displayed:

Example of a Chart with Asset Cost (Sum) by Category



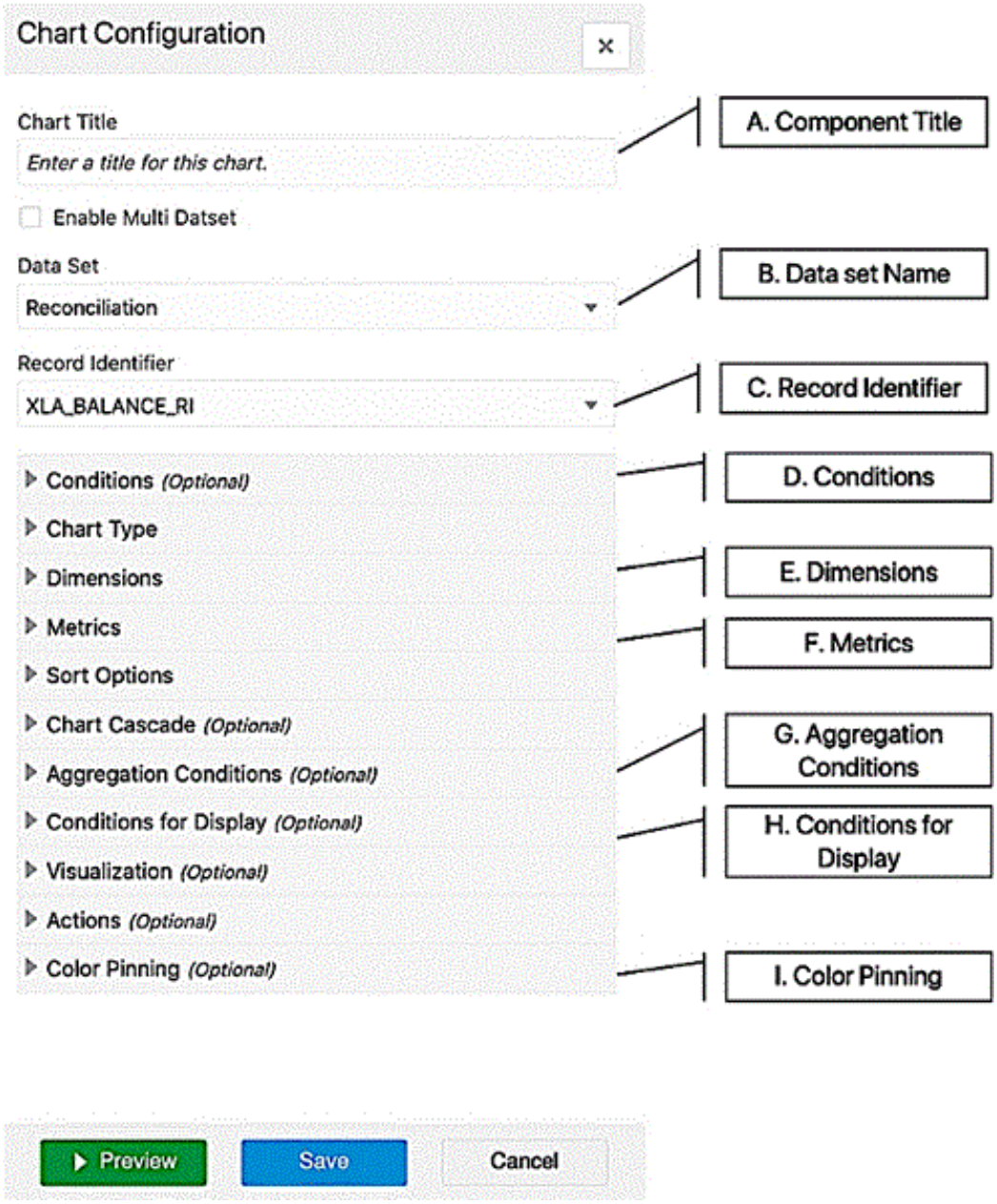
Common Configuration

Many components share common configuration options as follows:

- Component title
- Data set name
- Record identifier
- Conditions
- Dimensions
- Metrics
- Aggregation conditions

- Conditions for display
- Color pinning

Common Configuration Options



Configuration Details

The following table lists descriptions of all configuration options across different components.

Configuration Options

Configuration Option	Mandatory/Optional	Description	Example
Component Title (all)	Mandatory	In navigation components. Title must be Selected Refinements Available Refinements	
Component Title (all)	Optional	Dynamic/Static: in visualization component. Dynamic: displays the metric attribute (aggregation function) by dimension.	Chart: Asset Cost (Sum) by Category
Enable Data Set	Optional	Flag to enable configuration of multiple data sets.	
Data Set (all)	Mandatory	Select the data set display name	Assets
Record Identifier (all)	Optional	Used in case of entities in the data set at a different cardinality than the grain. When no record identifier is specified, records are displayed as-is. Affecting aggregation.	Asset Number
Condition (all)	Optional	Applies to both dimension and metric. Compound conditions with logical operators (AND/OR) applied among the conditions or among a group of conditions	((Record Type = 'I' AND Hold Count >= 0) AND (Discount Amount Available NOT NULL OR Second Discount Amount Available NOT NULL))

Configuration Option	Mandatory/Optional	Description	Example
Chart Type (chart)		Support different types of charts: Bar, bar/line, pie, donut, scatter and bubble Controls how bar, bar/line, and line chart is displayed (vertical/horizontal) Controls how stacked chart displayed (stacked/unstacked) Allow enabling zoom	Pie
Visualization (chart)		Chart Line Smoothing: Controls display of smoother line for line series Show Data Points: Controls display of data points on lines that are displayed on the chart For Pie and Donut charts: Allow runtime changes: Control to allow business users to change the threshold or number of dimensions + Add Threshold: Set the threshold value for each metric Other chart types: Number of Dimensions: Controls the number of dimensions displayed on the chart Order of Display: Controls the display of first or last dimensions after sorting order of display	
Visualization (tag cloud)		Show Metric Value: display the tag cloud term with relative weight Number of items: controls how many terms are displayed	

Configuration Option	Mandatory/Optional	Description	Example
Dimensions (all visualization components)	Mandatory	Qualitative data (string, strings, and date attributes Define the level of granularity that shows in the component	Major Category, Minor Category
Metrics (all visualization components)	Mandatory	Quantitative data Aggregated based on a given dimension. Support SUM/AVG/MIN/MAX/COUNT DISTINCT and COUNT in Aggregated Table. Only one metric displayed at runtime (except multi-metric chart).	Asset Cost (Sum)
Sort Option (Chart, Results Table, and Grid)	Optional	Enable sorting based on the dimension or metric defined. Allow runtime sorting in the chart and results table).	
Cascading (Chart, Tag Cloud)	Optional	When the data is refined to a single value for the dimension value, the component is updated to use a different dimension.	Major Category, Minor Category
Aggregation Condition	Optional	Allow having condition on the defined metric.	Asset Cost (Sum) >= 10000
Condition for Display	Optional	Control when the data is displayed on the component. Used when displaying the data within a context.	Book code (count distinct) = 1

Note: Oracle Enterprise Command Center Framework currently does not enforce validations on the component configuration at design time. Any invalid configurations will error out and be displayed in place of the component at runtime. Revisit the configuration and adjust accordingly.

The subsequent sections cover how UI components enable users to handle diverse scenarios and obtain several types of insights.

Navigation Components

Navigation components enable interactive navigation through the data without having prior knowledge of its distribution nor characteristics. Also, they allow searching for a specific keyword or term, apply filters, and reset applied filters.

Search Box

Oracle Enterprise Command Center Framework comes with search capabilities that allow users to search for a term within a particular data set.

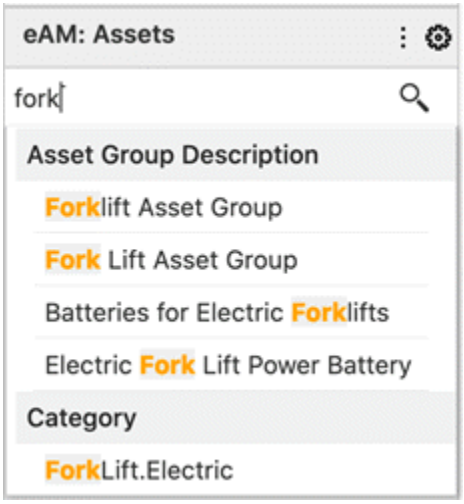
Search Box Overview

To perform a basic search, type your search term into the search box and Oracle Enterprise Command Center Framework will then list available matches in attribute values. If you select a value from the suggestion list then Oracle Enterprise Command Center Framework will execute a specific search on that attribute match. Alternatively, you can click on the magnifying glass (search icon) to retrieve all records containing this value regardless of which attribute matches your search criteria.

You can also switch between the configured data sets to search for available matches within the selected data set. The name of the selected data set will appear on top of the search component.

Introduced in V9, the Search Box and Selected Refinement components have undergone a redesign to conform with the updated Oracle design standards.

Search Box

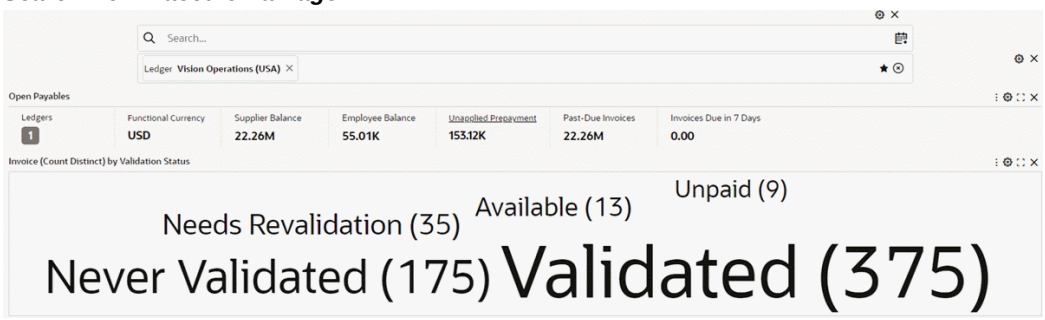


Oracle Enterprise Command Center Framework has advanced search capabilities in the search box such as wildcards, phrase and Boolean search. These types of searches are described below.

The search box can be dragged to the page to be included as a new component on the page or vice versa. The search component can also be added to the page from the 'Add Components' panel.

> **Note:** Only one search component per page is allowed and can exist either in the navigation panel or on the page but not at both the places.

Search Box Placed on a Page



Search Scope

Beginning with V11, designers can customize the search experience in an ECC dashboard to allow users to search only within the context of applied filters or search across everything regardless of applied filters.

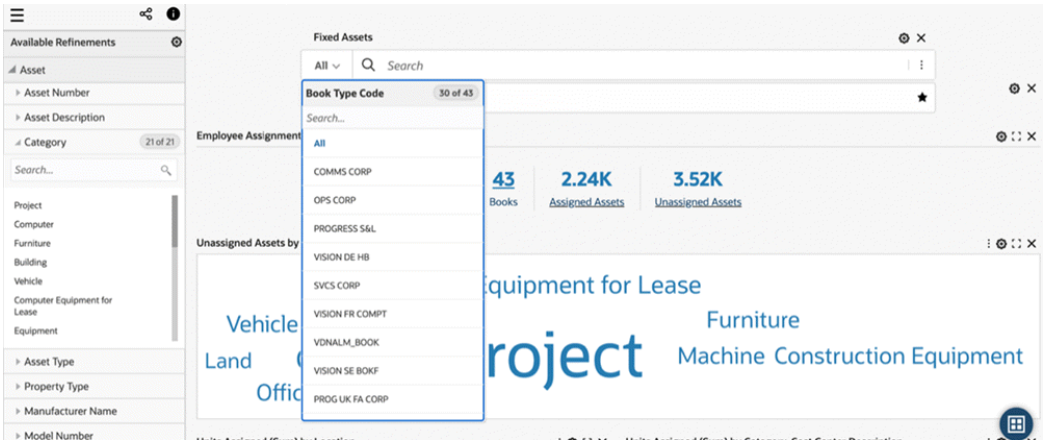
This is possible via the introduction of a new configuration element called "Search Scope," which has two options:

- Search Within: This option is the default setting, enabling users to search within the context of applied filters.
- Search Without: By toggling the search scope to "Search Without," users can search across all content regardless of applied filters.

Search Category

The Search Box feature supports the search category, which allows for searching within a specific attribute value.

Search Category in Search Box



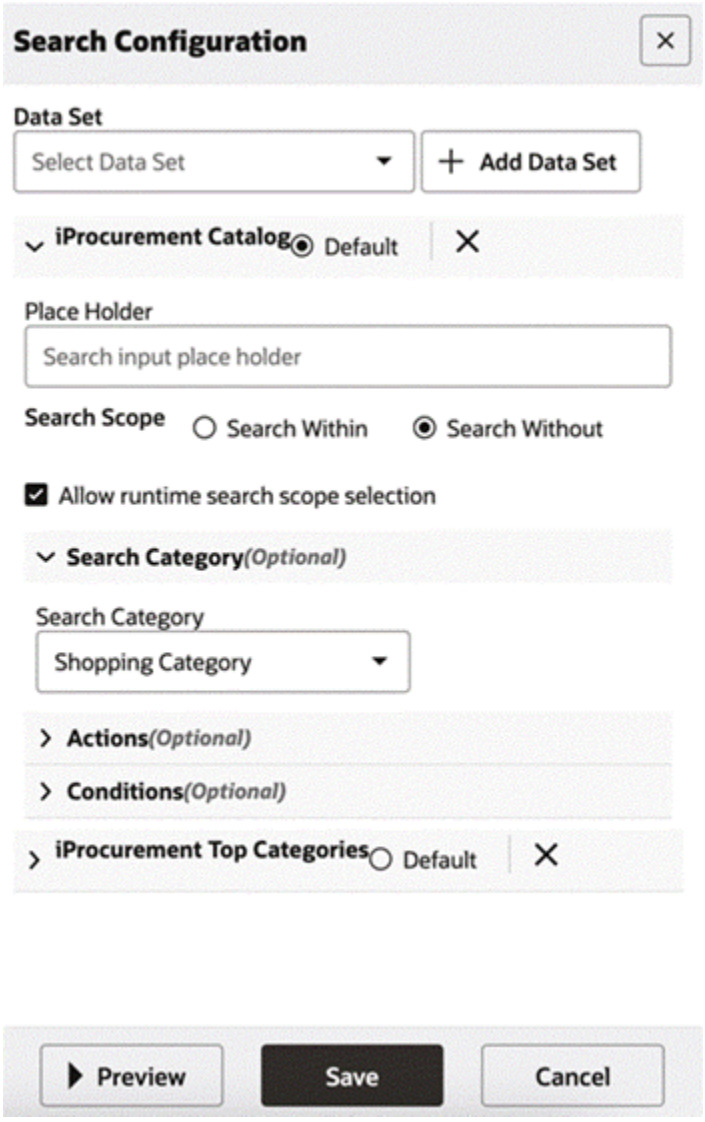
Search Configuration

Configuration Options for Search Box

Option	Description
Data Set	Multiple data sets can be added. The configuration has to be repeated for each data set.
Default	Used to configure the data set that is selected by default for the search component at runtime. Only one data set can be configured as the default.
Place Holder	String defined to act as a placeholder inside the search bar. Default is: 'Search ...'.
Search Scope	Control to configure search experience to be either "Search Within" or "Search Without".
Allow runtime search scope selection	Enabling this checkbox allows business users to define search behavior at runtime; that is, users can set the search scope to be either "Search Within" or "Search Without".

Option	Description
Search Category	Select the attribute for search. In the runtime, all the values for this attribute are available in a list for the user to select. They can also be searched from an embedded search box.
Actions	Select Action Type Refinement: Applies refinement to the page using search value Hyperlink: Navigates to the page provided after applying refinement with search value

Search Box Configuration



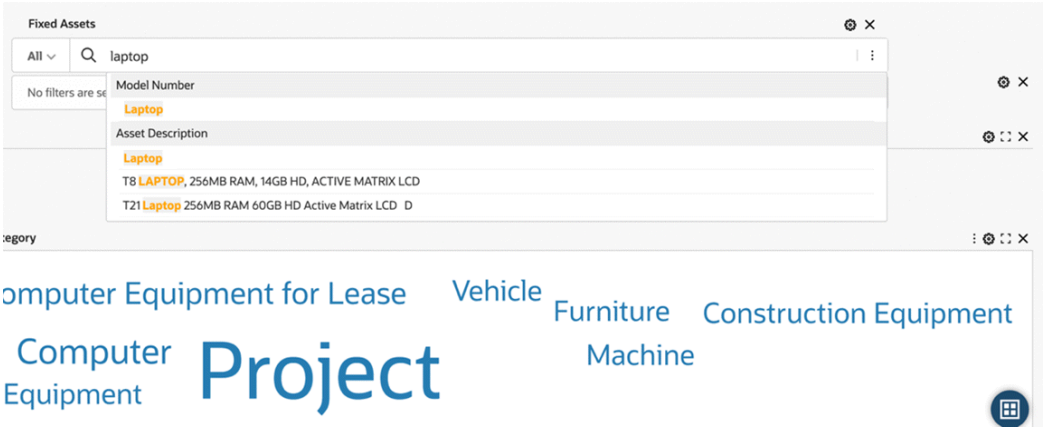
Search Behavior

Various types of searches and the associated behaviors are described below.

Value Search

Upon Selecting an attribute value from category and further selecting another attribute value from suggestion list of search adds both category attribute value and selected attribute value as refinements in selected refinements.

Value Search



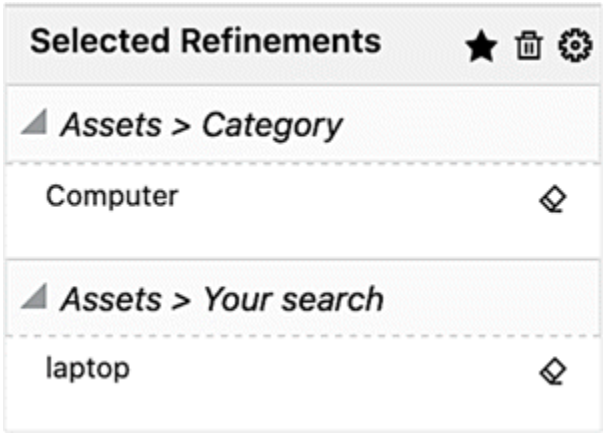
Beginning with V6, search suggestions of a value search are subject to a control in metadata. Only the attributes with flag "Search Suggest List?" are part of value search suggestions.

Record Search

Introduced in V6, record search exhibits the same behavior as value search. Search queries are added as refinements by clicking on the magnifying glass. Search queries are applied as refinements to the page under 'Your Search' accordion.

Search Query Example

'Your Search' Accordion Example



Phrase Search

Phrase Search allows you to search for an exact sequence of terms using quotation marks (`" "`). For example, searching for `phillip taylor` returns only records with `phillip taylor` (case-insensitive).

Wildcard Search

Wildcard or partial search is used when searching for a term where you only know a few letters. By default, when you start typing your search term in the search box, a trailing wildcard (*) is implicitly added at the end of the word. For example, a search for `work` returns all values with terms that start with `work`; for example, `WORKSTATIONS`.

Also, you can use an asterisk (*) or percent sign (%) at the beginning of the search term, and the results match any text contains the characters between the search operator, even if they occur in the middle of a word. For example, with a search for `*work*` or `%work%` , all values that have 'work' are returned in the search results. The (*) or (%) symbols perform a multiple character wildcard search that looks for 0 or more characters. Thus, a search for `*work*` returns values such as `PC WORKSTATION`, `NETWORK`, and `WORKSTATIONS`.

Boolean Search

You can include logical operators in the search to set more precise search logic based on the operators listed below: Note: Boolean operators must be in capital letters only: AND, OR, and NOT.

> **Note:** Boolean operators must be in capital letters only: `AND`,`OR` , and `NOT`.

The following table describes these operators:

Boolean Search Operators

Operator	Purpose	Example Usage and Results
AND	Returns results with all specified terms.	`cip AND addition` Returns results with both 'cip' and 'addition'
OR	Returns results with any specified terms.	`desktop OR laptop` Returns results with either 'desktop' or 'laptop'
NOT	Negates the following term (Will not retrieve records that have the unwanted keyword).	`desktop NOT monitor` Returns results with 'desktop' but not 'monitor'

Operator precedence is determined in the following order:

1. Any sub-expressions in parentheses are evaluated first
2. `NOT` is evaluated before other operators
3. `AND` is evaluated after `NOT`
4. `OR` is evaluated after `AND`

Expressions

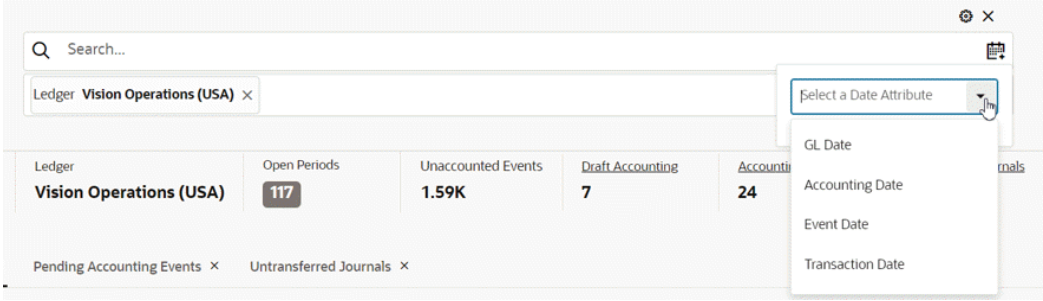
An expression is used to build a more complex search query. You can combine keywords with `AND`, `OR`, or `NOT`. Use parentheses to determine the relationship between operators when more than one operator is used. For example, a search for `(computer OR desktop) NOT monitor` returns records that contain both the words `computer` and `desktop`but do not contain `monitor`.

Quick Date Filter

Introduced in V12, the Quick Date Filter feature significantly enhances user experience by allowing users to swiftly select date filters without navigating to available refinements. This feature enables users to select date attributes, choose from predefined time ranges, and utilize relative date ranges such as "last 7 days" or "previous month."

Example of Quick Date Search Icon and Tooltip

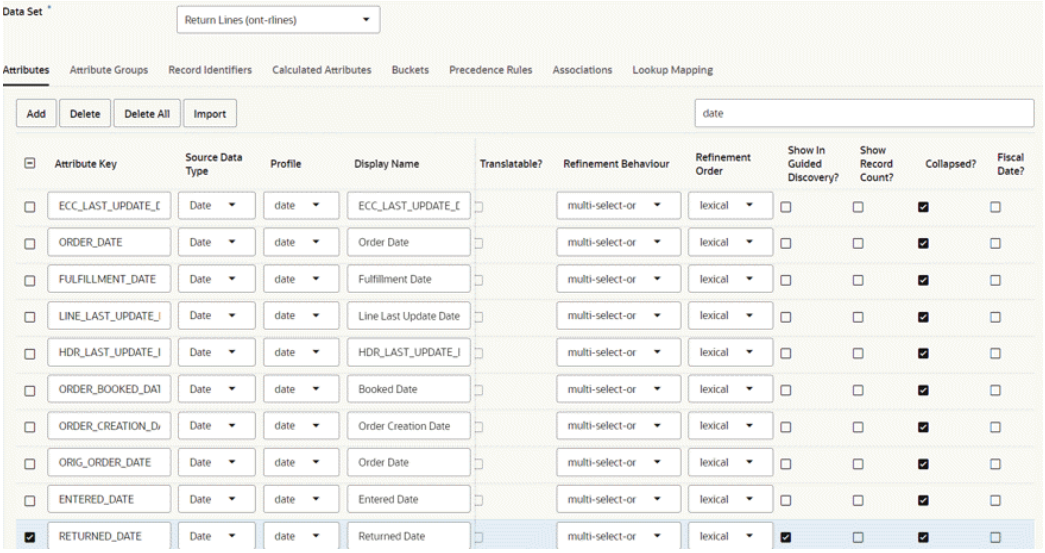
Example of Quick Date Filter with Expanded List



> **Important:** Note that Quick Date Filter is not applicable to date subsets or fiscal years.

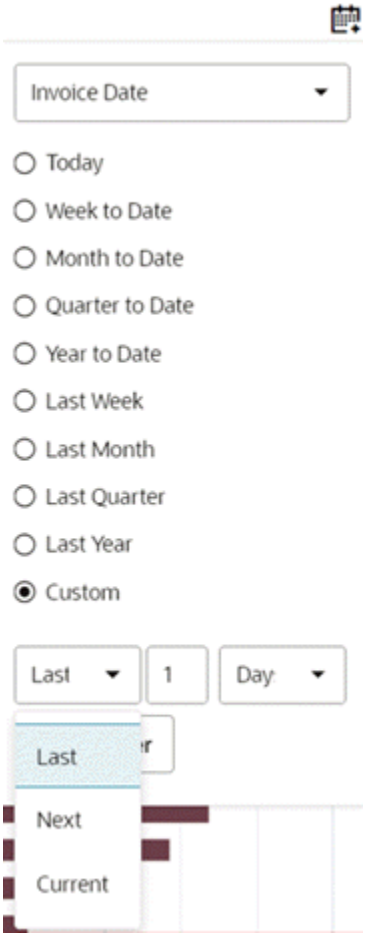
The implementation of this feature requires no configuration changes and is seamlessly embedded within the search box. The date list is generated based on date and date-time attributes marked with the "Show in Guided Discovery" flag in the metadata.

Example of Data Set Page, Attributes Tab, with "Show in Guided Discovery" Selected for RETURN_DATE



In cases of multiple datasets, the date list is derived from the default dataset configured under search settings. Users can select from a variety of quick filtering options and even apply custom relative date filters. The levels available are Days, Weeks, Months, Quarters, and Years. Additionally, users have the flexibility to choose filtering at the levels of Hours and Minutes for date-time attributes.

Options to Choose "Last," "Next," and "Current" for a Date



Once applied, quick date filters are added to selected refinements as regular date filters, allowing users to amend them and switch between list, range, and relative range type date filters as needed.

To utilize this feature, ensure that the date or date-time attribute intended for quick filtering is designated as "Show in Guided Discovery?" in the metadata as described above.

Selected Refinements

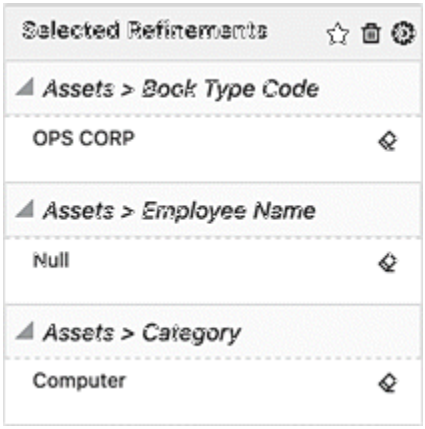
Selected Refinements display all values that the user has selected to filter the dashboard, organized by attribute name and data set. The captured filtered can come from user interaction with any of the Oracle Enterprise Command Center components that allow refinements such as search box, tag clouds, charts, and so on.

The Selected Refinements component can additionally control associative filtering behavior and make the dashboard sensitive to refinement state coming from directly or indirectly associated data sets. This is done by specifying the associated data sets in the

configuration.

Beginning with V6, a user can view the meaning or values of filters applied in selected refinements by hovering on them. This feature is controlled by Lookup Mapping in metadata. Oracle Enterprise Command Center Framework leverages a mapping between the attribute code and the attribute description to show the description in a tooltip when the hovers on the attribute code in Selected Refinements.

Selected Refinements



Selected Refinements have the look and feel options described in the following table:

Selected Refinements Look and Feel Options

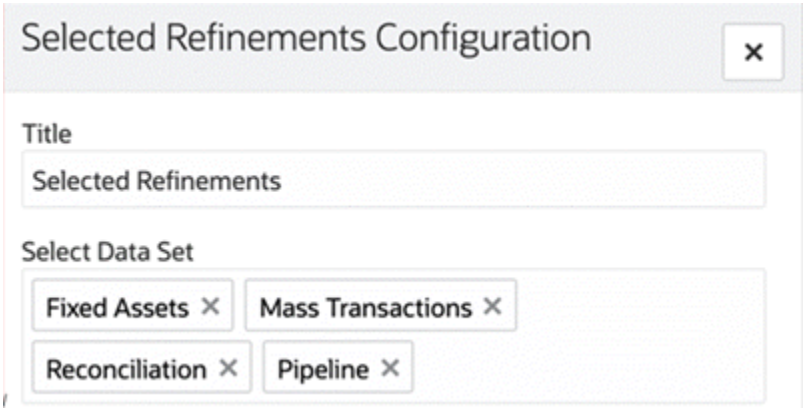
Page Layout	Selected Refinements
Default (Side Navigation)	Displayed inside the navigation panel. If the user collapsed the side navigation panel, a funnel icon is displayed at the top of the page.
Side Navigation Collapsed	Displayed as a funnel icon. If the user opens the side navigation panel, selected refinements are displayed at the top of the page.
No Side Navigation	No selected refinement on the page.

Configuration

Configuration Options for Selected Refinements

Option	Description
Title	Default title "Selected Refinements"
Data Set	Supports more than one data set selection for honoring association and applied filters between defined data sets.
Override Applied Filters	With this flag enabled, if the user applies filters from a different data set, then filters from any other data sets are deleted in selected refinements.

Selected Refinements Configuration



Breadcrumbs

Introduced in Version 4, the Breadcrumbs feature is an intuitive representation of the selected refinements as a trail of filters on the dashboard page. This feature is configured along with the search box on page. It emphasizes the sequence of the steps in the path the user has chosen to arrive at the current state of the dashboard. The Breadcrumbs component can be added in two ways:

- Dragging the selected refinements section to page area
- Adding a new component: 'Selected Refinements' from the components list

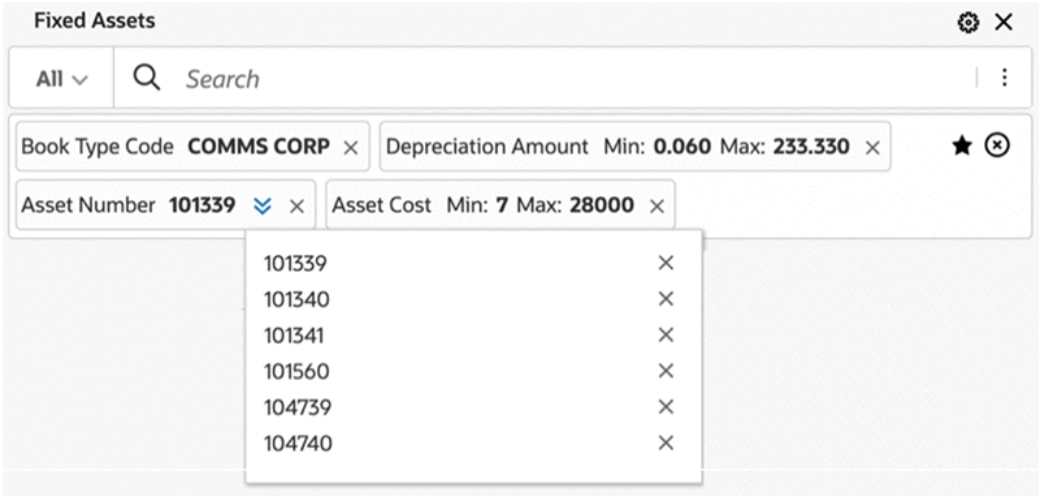
The Breadcrumbs component shows the data set name on top of each filter. It lists all types of filters available in selected refinements. User can delete individual filters or

clear all filters from the breadcrumbs.

If there are more than three filters of the same attribute, all the filters are grouped and collapsed displaying an icon. Clicking on the icon displays all the filters of the attribute. All the filters for such attributes can be removed at once. Users can hover over a refinement to view the data set name.

Beginning with v9, the breadcrumbs for selected refinements was re-designed. A user can hover over a refinement to view the data set name.

Example of Breadcrumbs for Selected Refinements



Beginning with V7, the Breadcrumbs component has been improved to enhance user discovery capabilities by allowing the user to replace or apply additional filters from the selected refinement, inheriting all the refinement behavior as in available refinements.

If you click on the attribute name, then you can select another value and apply it as a filter on top of the existing one from the same attribute.

If you click on the attribute value, then you can select another value and replace it with the existing one from the same attribute.

> **Note:** This behavior is applied to multi-assigned attributes with `OR` and `AND`. If you filter by one of the context attributes (single assign) in the dashboard, like a ledger, operating unit, or inventory organization, and then click on the attribute name or value, the applied filter will be replaced by the new value.

If you applied a date range filter, then you can click on the attribute name or attribute value and replace the existing filter with the same attribute.

Also, if you applied a date range filter, and then clicked on the attribute name or value and switched from a range to a list or from a relative range to a list, and next filtered by

one value or more from the list, then these filters will be applied on top of the existing range filter applied from the same attribute. The new filter will be added in different accordions because it is based on different operators (in accordance with standard behavior in ECC).

If you applied a numeric range filter, you can click on the attribute name or attribute value and replace the existing filter with the same attribute.

Beginning with V10, Breadcrumbs for Selected Refinements has been enhanced to allow users to make quick selections simply by clicking on the respective attribute. Users can also make multiple selections by clicking on the respective checkboxes, followed by clicking on the **Select Filter** button. Negative refinements are also simplified as users can click on the negative refinement icon for an attribute to negatively refine the dashboard.

Saved Search

Oracle Enterprise Command Center Framework provides an option to save the frequently applied filters or preferred filters as saved searches for allowing users to reuse them. All saved searches are context-sensitive to the page and are part of the search suggestions. The list of saved searches can be obtained when focused on the search component. Saved searches are searchable by their title, filter attributes and filter values.

Three types of saved searches are available for the users: Seeded, Public, and Private. Seeded saved searches are published along with the product, Public saved searches are created by admin users and all the saved searches created by users are called Private saved searches. Private saved searches are accessible only by the users who created them, whereas public saved searches are accessible by all the dashboard users.

For more information, see Saved Search, *Oracle E-Business Suite User's Guide*.

Available Refinements

The Available Refinements feature enables interactive navigation through the data without having prior knowledge of its distribution nor characteristics.

As a user interacts with available refinements components or perform filtering operations from other components on the dashboard, Available Refinements dynamically updates the attribute list to show relevant attributes and attribute refinements. This navigation is data-driven and supports progressive disclosure of additional attributes as appropriate according to the user's navigation path through the data. Data rendered in the attribute value list also honors refinement state and can shrink, expand or be removed based on which subset of data user is exploring.

Available Refinements supports displaying attributes in a grouped list to reduce clutter.

Available Refinements also supports switching between data sets to apply relevant filters from the data sets. The name of the selected data set will appear in the header of the available refinements section.

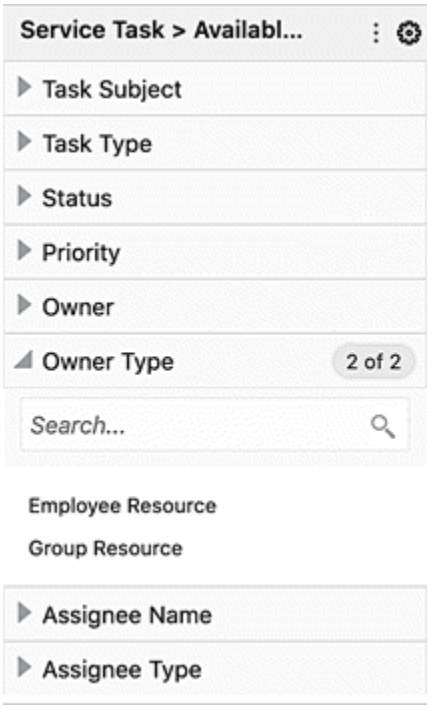
Oracle Enterprise Command Center Framework supports advanced scrolling functionality by limiting the height of refinements section by the height of the dashboard. Any expansion of refinements due to the addition of selected refinements or expansion of accordions in available refinements will be within this height.

Descriptive flexfield (DFF) attributes can be accessed from available refinements along with customer defined labels for DFF attributes. Oracle Enterprise Command Center Framework allows progressive disclosure for context-sensitive attributes.

Beginning with V6, users can see easily understandable value set descriptions in available refinements instead of simply value set codes.

The Available Refinements feature also supports filtering the dashboard with time-level precision for date-time attributes. A user can select a time range along with a date range.

Available Refinements

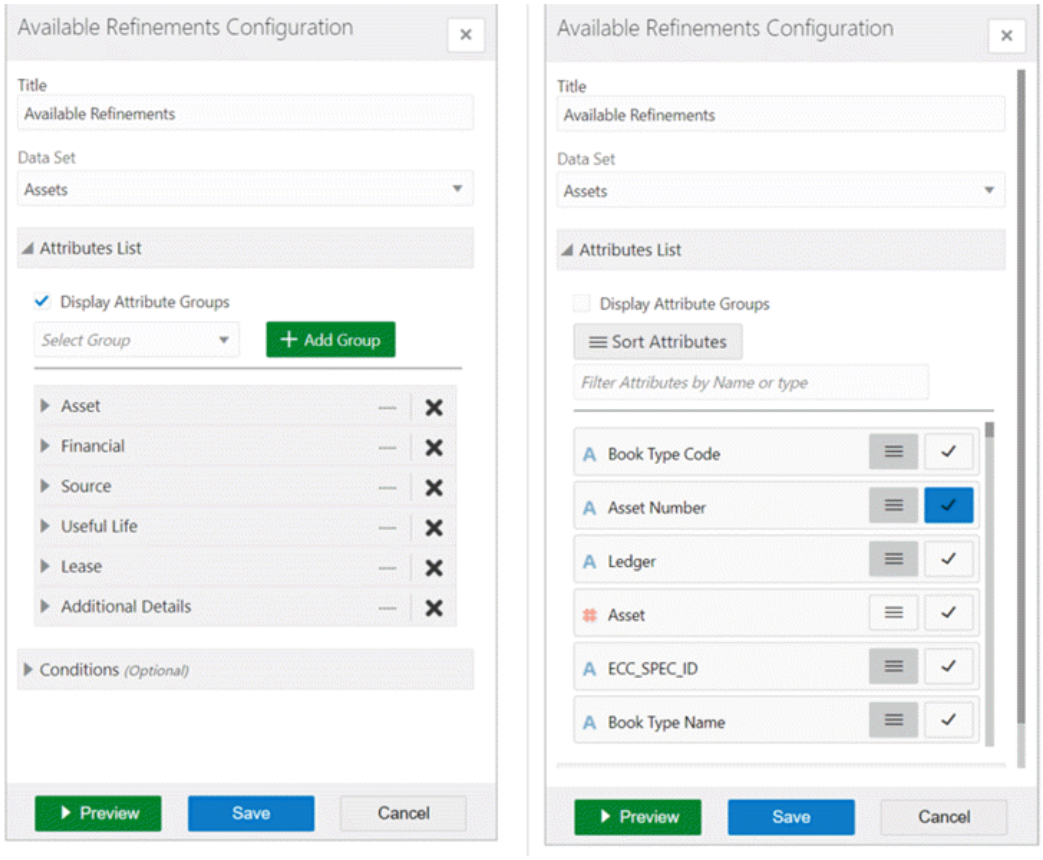


Configuration

Configuration Options for Available Refinements

Option	Description
Title	Default title "Available Refinements".
Data Set	Multiple data sets can be added.
Default	Only one of the data sets can be selected. Available refinements select that data set as the default data set at runtime.
Attribute List	Display Attribute Groups: this checkbox allows selecting and displaying attribute groups. Attribute list: allows controlling of attribute display in the component, and how the attribute is displayed (list/range). These controls are supported in both attributes flat list and attribute groups.

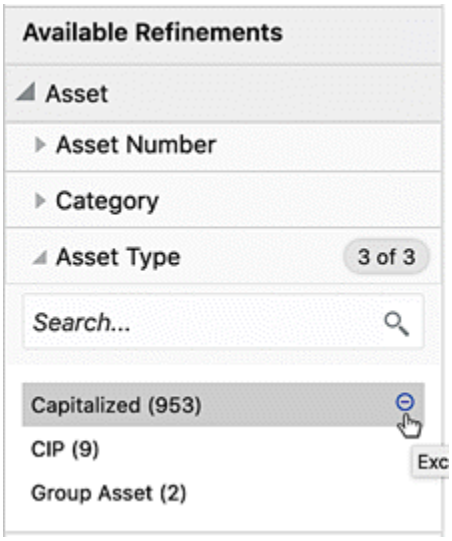
Available Refinements Configuration



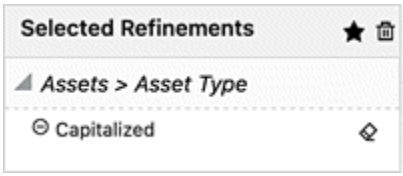
The Negative Refinements feature allows users to refine the data by filtering out the selected values. Such refinements are displayed in selected refinements with an **Exclude** icon indicating them as negative refinements.

To apply a negative refinement, you can click the **Exclude** icon. The icon is visible only when hovered on the attribute value or navigated to using the keyboard.

Example of a Applying a Negative Refinement

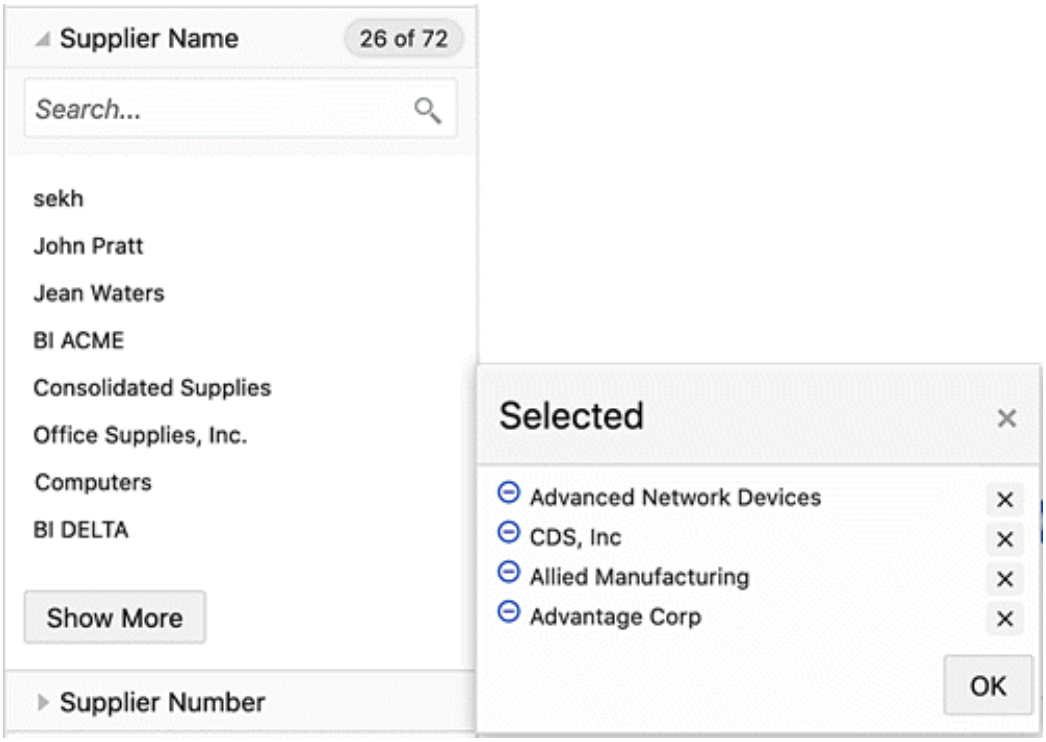


Example of an Applied Negative Refinement



Also, more than one value can be selected using the **Exclude** icon for all the desired exclusions.

Example of Multiple Negative Refinements



Example of Multiple Negative Refinements Applied

If a search is made, the **Exclude All** button excludes all the search results.

Negative refinements change the refinement behavior of multi-select AND attribute to multi-select OR and vice-versa.

Behavior Matrix for Refinement Selections

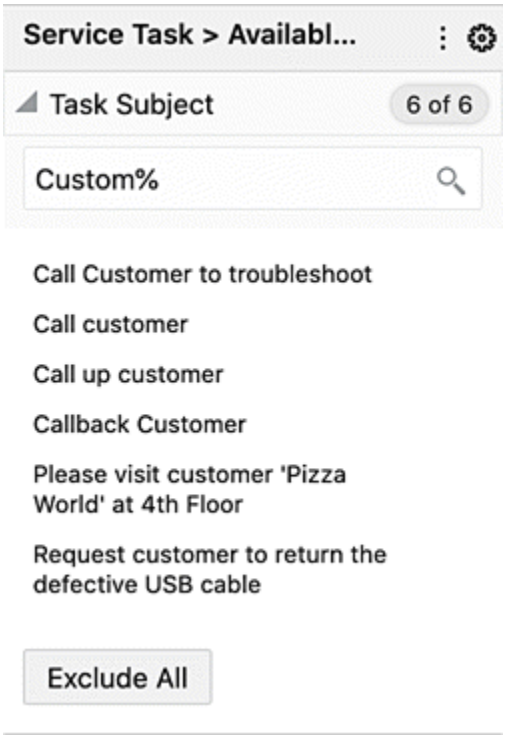
Negative Refinement Selection	Single Assignment, Positive	Single Assignment, Negative	Multiple Assignment, Positive	Multiple Assignment, Negative
Single Selection	=	< >	N/A	N/A
Multiple Selection OR	N/A	N/A	OR	NOT X AND NOT Y
Multiple Selection AND	N/A	N/A	AND	NOT X OR NOT Y

Like Search

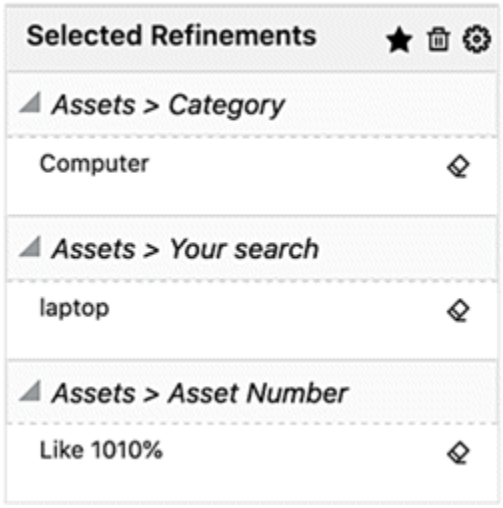
Available refinements support LIKE search for attribute type: 'string' and search queries using operators such as: '*/%' - for zero or more characters and '?' - for one character allowing the user to perform searches with partial matches or exact matches. Search results are immediately applied as refinements in selected refinements.

Clicking the magnifying glass icon performs a *LIKE* search even when no partial search operators are included in the search query. A partial search operator is implicitly added at the end of the search query if no operator has been mentioned.

Like Search in Available Refinements



Like Search in Selected Refinements



Additional Features for Filters

Range Filter

Beginning with V7, a user can switch between the range display and list display for date and numeric attributes in the available refinements and breadcrumb features.

Also, a slider is available in the numeric attribute to improve user usability for range filters.

Relative Date Filter

Beginning with V7, a Relative Date Filter allows the business users to filter on the dashboard based on a sliding window of time. Users can also take advantage of the default saved search with the relative date.

Users can apply time-based filters to any date and DateTime column in the dashboard with the relative date filter. For example, users can use the relative date filter to show only sales data that has occurred within the last 30 days (calendar months). From the available refinement; the user can filter by relative date filter from any date attribute by clicking on the relative date icon.

The default filter is "Today" and the user can switch between the different variations of relative dates. Also, the user can specify a relative date period as either an explicit number of past or future time units (for example, 2 years) or specify a previous period. For example, Year To Date, which includes data from 1-January this year to the current date, and Month To Date, which includes data from the beginning of the month to the current date.

Visualization Components

Oracle Enterprise Command Center Framework has a set of graphs and charts that provide a powerful way of summarizing and presenting data that are critical in decision making. The user can thus find insights, detect outliers, filter the data directly from the charts, and drill down to a deeper level of detail. The following sections describe these components.

Summarization Bar

The Summarization Bar (also called the Summary Bar) allows users to get their footing into a particular business area by viewing metric or dimension values that summarize important aspects about the business area covered by the dashboard.

In V5, the summarization bar also displays abbreviated numbers for values of flag and metrics above 1000. The abbreviation is language-sensitive.

An entry in the summarization bar can be one of the types in the following table:

Entry Types for a Summarization Bar

Summary Item	Description
Metric	Displays the value of a specific metric, such as total sales or average profit. Can be used to navigate to a destination dashboard or tab and optionally invoke a refinement action.
Dimension	The dimension value associated with either the top or bottom value of an associated metric value, such as the product category with the highest total sales.
Flag	Flags display the count of the configured dimension. Additional metrics are displayed in the pop-up.

Summarization Bar

Configuration

The general configuration options and item-specific configuration options are listed below.

General Configuration

General Configuration Options for Summarization Bar

Option	Description
+ Add Summary Item	Click the drop-down arrow beside + Add Summary Item and select the type of summary item to add from the list.
Title	Summary item title
Data Set	Select data set

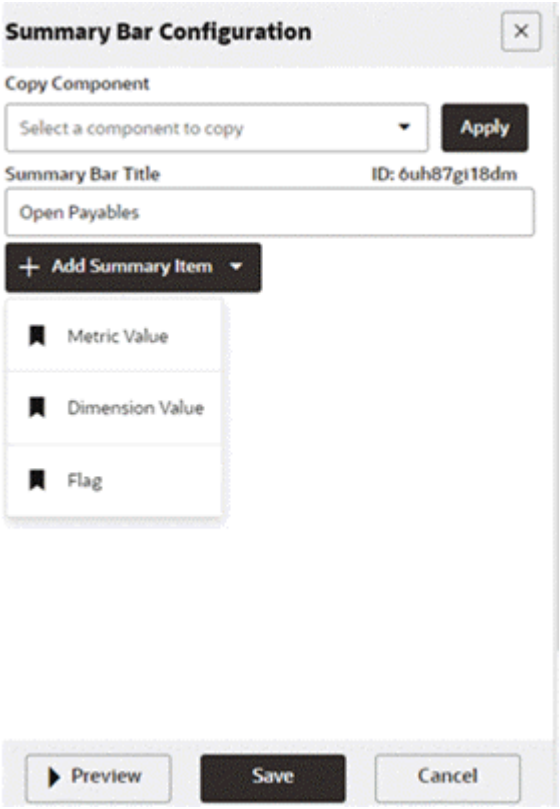
Notes:

- To determine the order for displaying summary items, drag each item in the list to the desired order.
- To delete a summary item, click its 'x' icon.

Configuration Steps for Summary Bar

1. Enter the summary bar title and add Flag as a summary item.

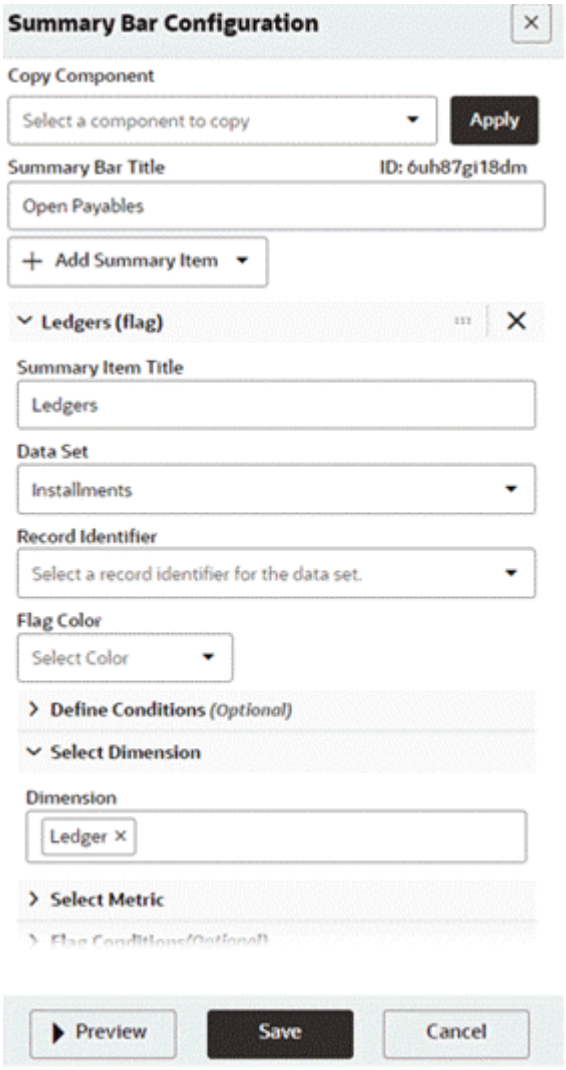
Example of Defining the Summary Bar Title and Adding Flag as a Summary Item



2. Enter the following:
   - Define the flag title.
   - Define the data set name.
   - Enter a record identifier if needed.
   - Enter a flag color if needed.

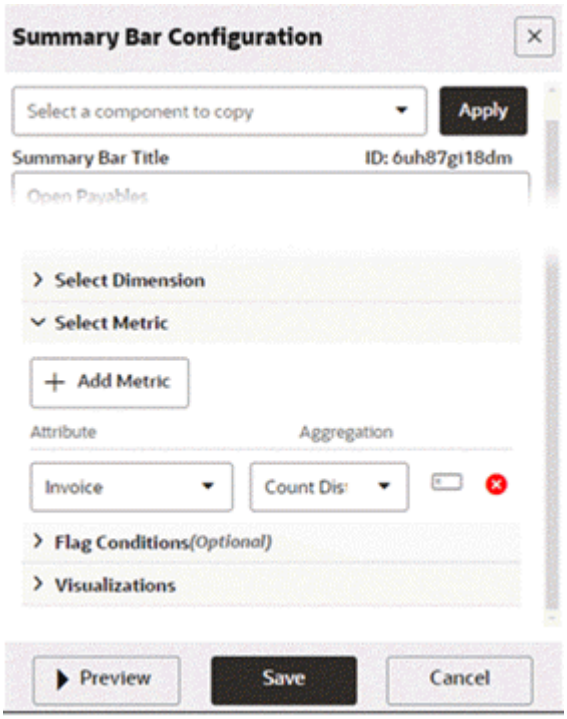
- Define conditions if any.
- Select the dimension.

Example of Additional Configuration Fields for a Summary Bar



3. Define a metric (optional).

Example of a Metric for a Summary Bar

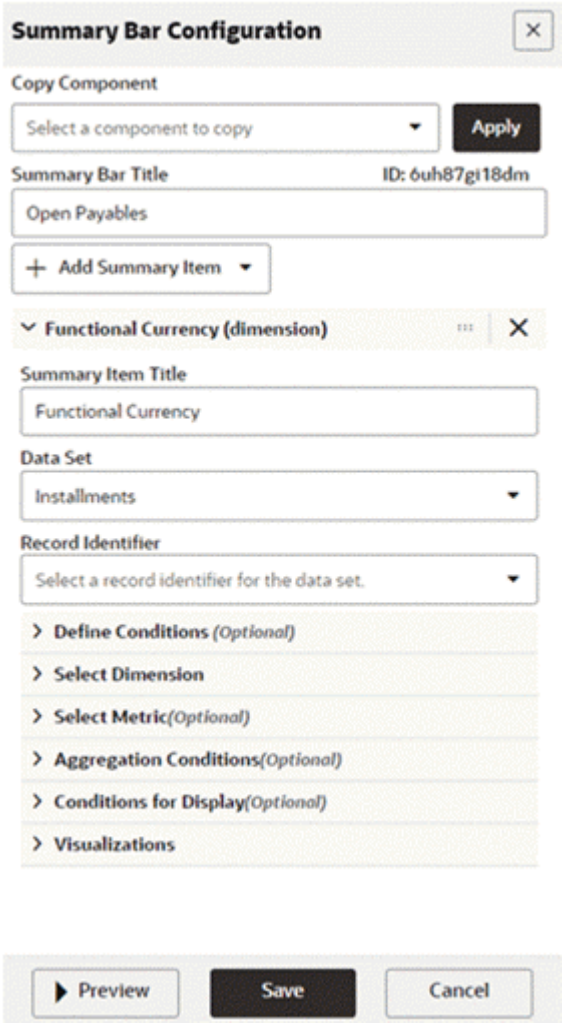


4. Add another summary bar item - Dimension.

   Define the summary item title.

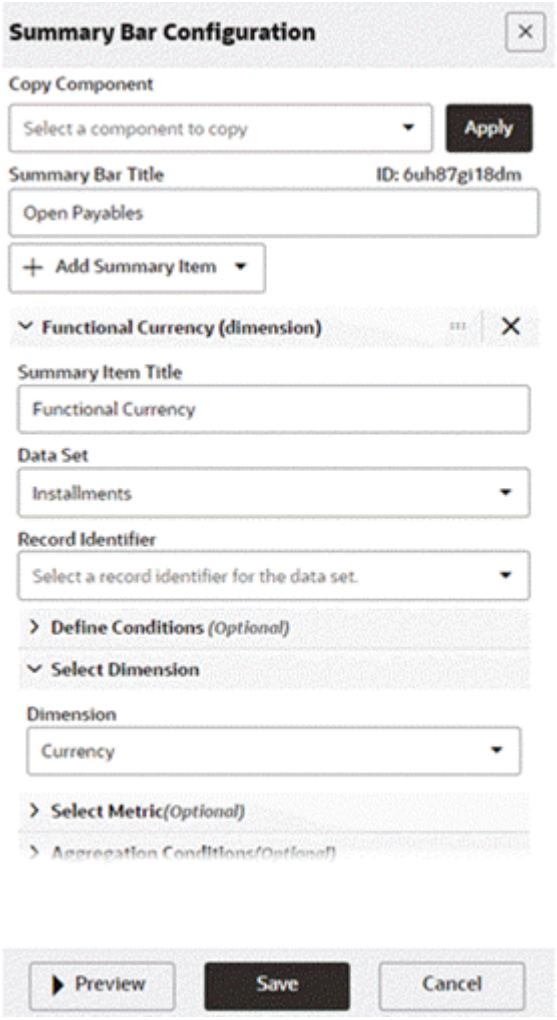
   Define the data set.

Example of Defining Another Summary Bar Item



5. Select the attribute to be used as the dimension.

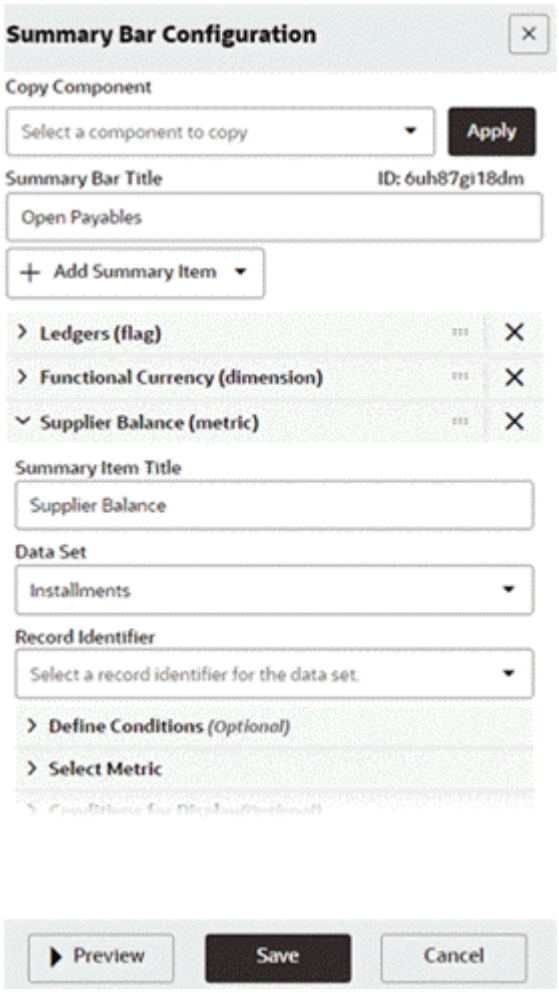
Example of Selecting an Attribute as a Dimension



6. Add a new summary bar item - Metric.

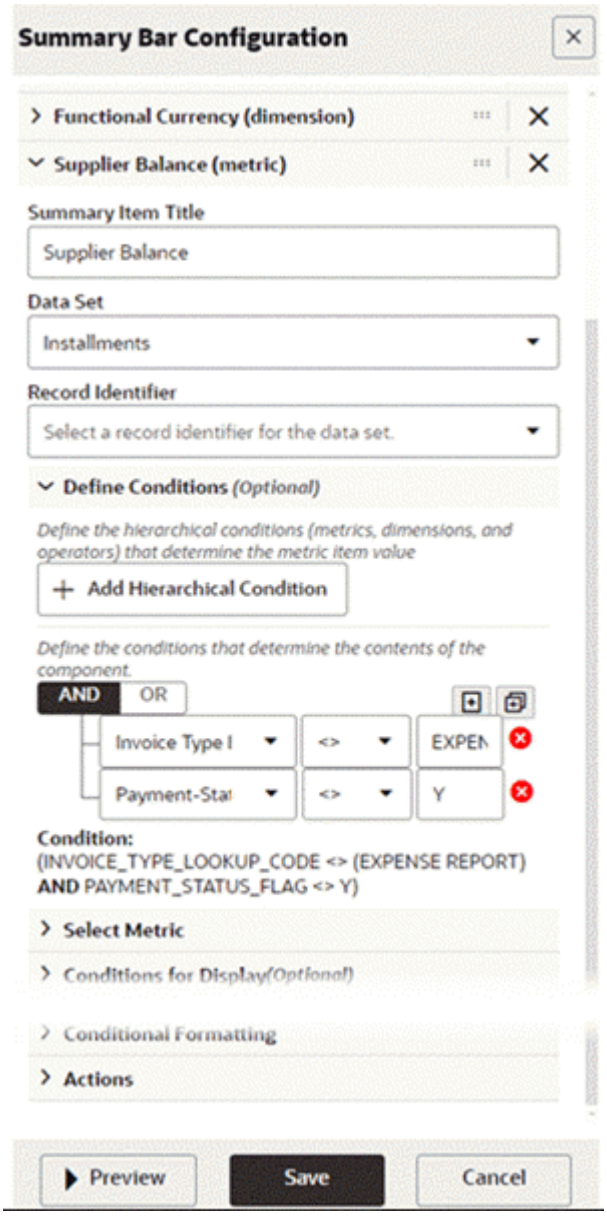
   Define a title and data set.

Example of Adding Another Summary Bar Item (Metric)



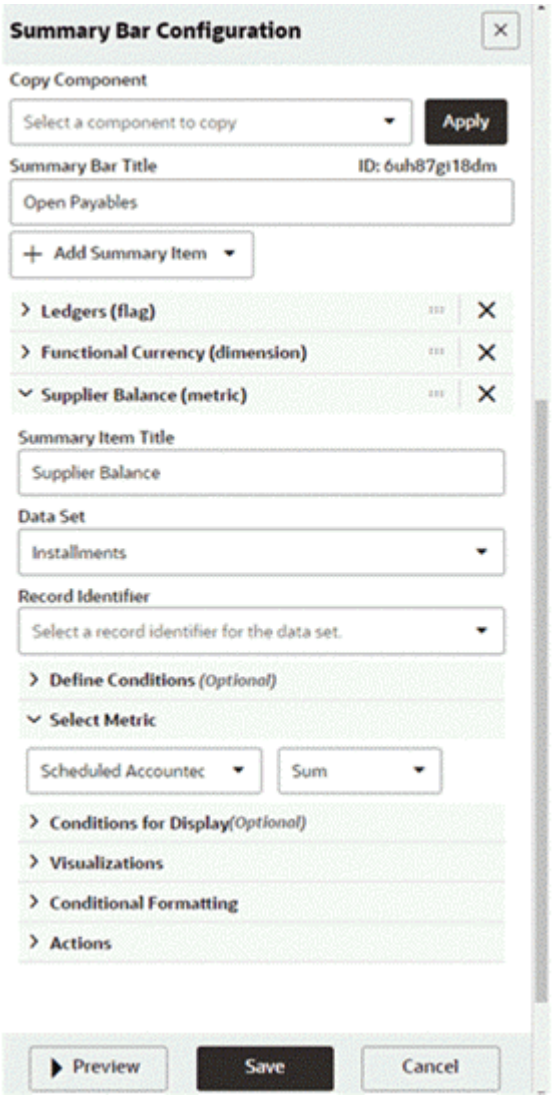
7. Add conditions as required.

Example of Defining Conditions for a Summary Bar



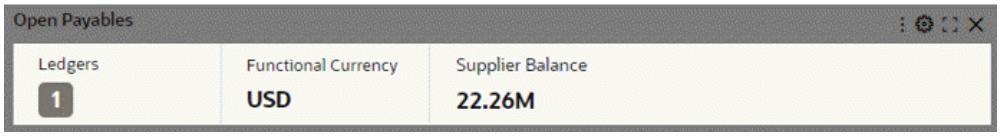
8. Define metrics.

Example of Selecting a Metric



9. Preview the summary bar.

Preview of a Summary Bar



Specific configuration per item

Visualization:

Visualization Options

Option	Description
Enable Tooltip	Allow adding a tooltip to describe the summary item.
Tooltip Text	Use the `{Value}`token to represent the value.

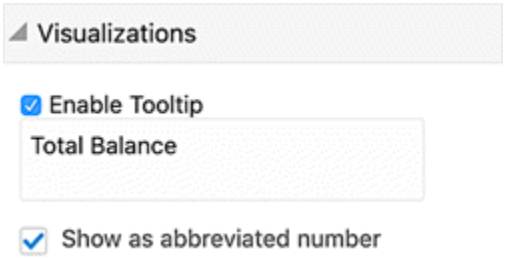
Abbreviated numbers can be shown, with the following characteristics:

- Supported for flag and metric
- Allows display of abbreviated numbers instead of actual values
- Actual values are displayed as a tooltip with any tooltip text beneath it
- Values above 1000 are abbreviated till displaying till two decimal points
- The abbreviation is language-sensitive

Visualization Option for Abbreviated Numbers

Option	Description
Show as abbreviated number	From Visualizations accordion, Enable Show as abbreviated number

Visualization Configuration Option



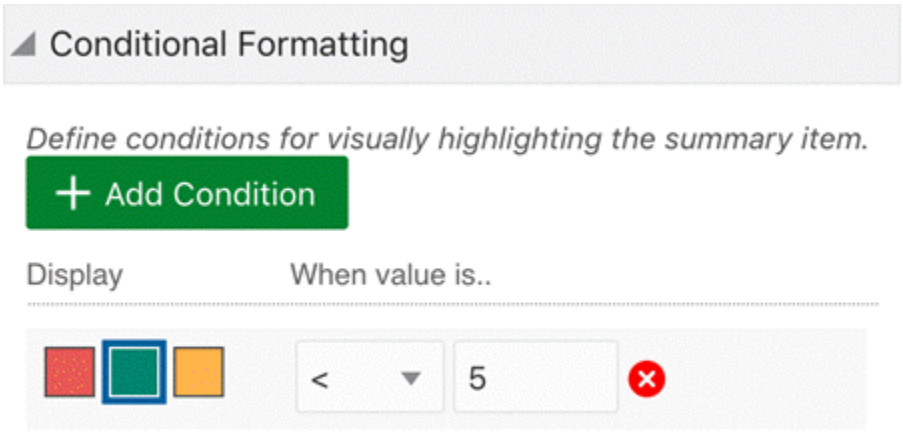
Conditional Formatting

- Supported only in the Metric item
- Enables displaying the metric values in different colors based on specific conditions

Conditional Formatting Options

Option	Description
Add Condition	Add one or more conditions
Formatting	• Choose a color to display upon meeting the condition • Select a mathematical operator to use • Enter the appropriate value in the Value field • Enter the minimum and maximum values if the 'between (b/w)' operator is chosen • Delete a condition using the Delete icon

Conditional Formatting Configuration



Conditional Formatting at Runtime

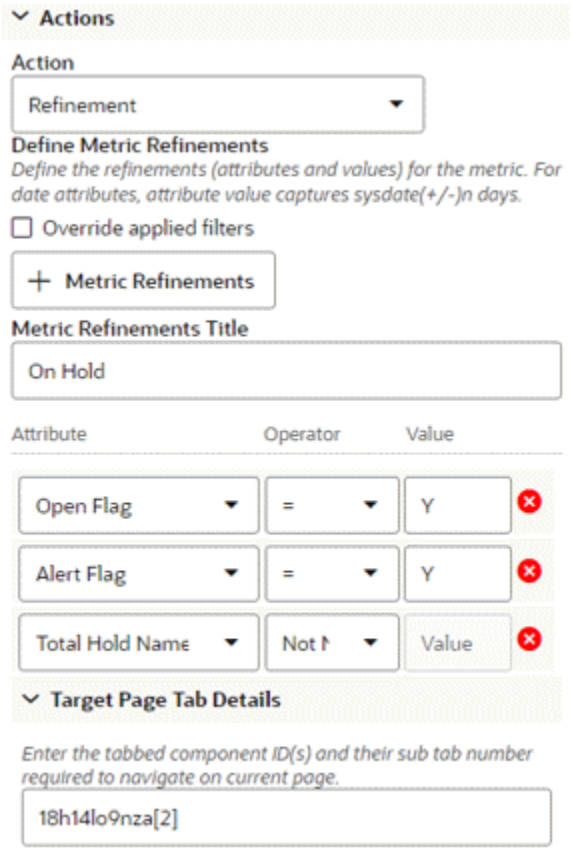
Action

- Supported only in the Metric item.
- Allows users to click the metric value to navigate to a different dashboard or apply a refinement action.
- Introduced in V12, aliasing is not supported in metric refinements. This is an optional parameter. If not specified, the component title will be used for aliasing. This feature enhances user experience by showing a meaningful name in the selected refinement. Upon hover, user can find out details of the refinements applied under the metric refinement.

Action Options

Option	Description
Action	From the action type list, select the action.
Refinement Action	• Click +Metric refinement. • Add the refinement action. • To navigate to a specific tab in a tabbed component container in the same dashboard, enter the tab component ID and the subtab number. • Note: Tab component ID generated and displayed in the tab component configuration. • Note: Metric refinements cannot apply a compound condition as refinements.

Metric Refinement Action Configuration



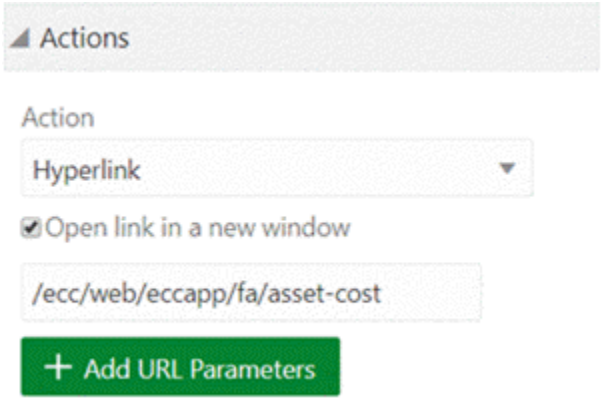
Metric Refinement in Runtime

Action Options

Option	Description

Hyperlink Action	• When users click the value, they navigate to the specified URL.
	• A hyperlink can be to another dashboard in the same application or an external URL
	• URLs can include attribute values as parameters.
	• To open a new browser window for the destination dashboard, select the Open link in a new window checkbox
	• Enter the hyperlink destination in the URL field.
	• To add attribute values to an external URL(optional):
	  • Click + Add URL Parameters.
	  • In the attribute list, select parameter attribute(s)
	  • The attributes are inserted as parameters in the URL, where the parameter name is the attribute key, and the parameter value is `{ID}`, where `ID` is the ID for that attribute.

Hyperlink Action Configuration



Flag Condition

- Use the Flag Conditions to add a condition that must be met for the flag summary

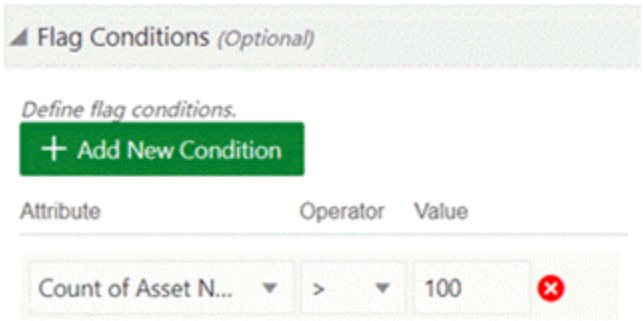
item to display.

- A flag summary item is only displayed on the end user view if at least one value matches the condition defined.

Flag Condition Options

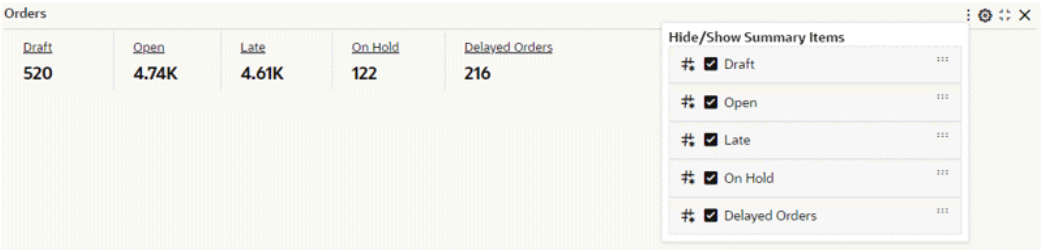
Option	Description
+ Add Condition	• Select a predefined metric • Operator • Value

Flag Condition Configuration



Beginning with V10, users can personalize the Summarization Bar component by including, excluding, or reordering items using the runtime options window. For more information, see User Personalization, page 7-250.

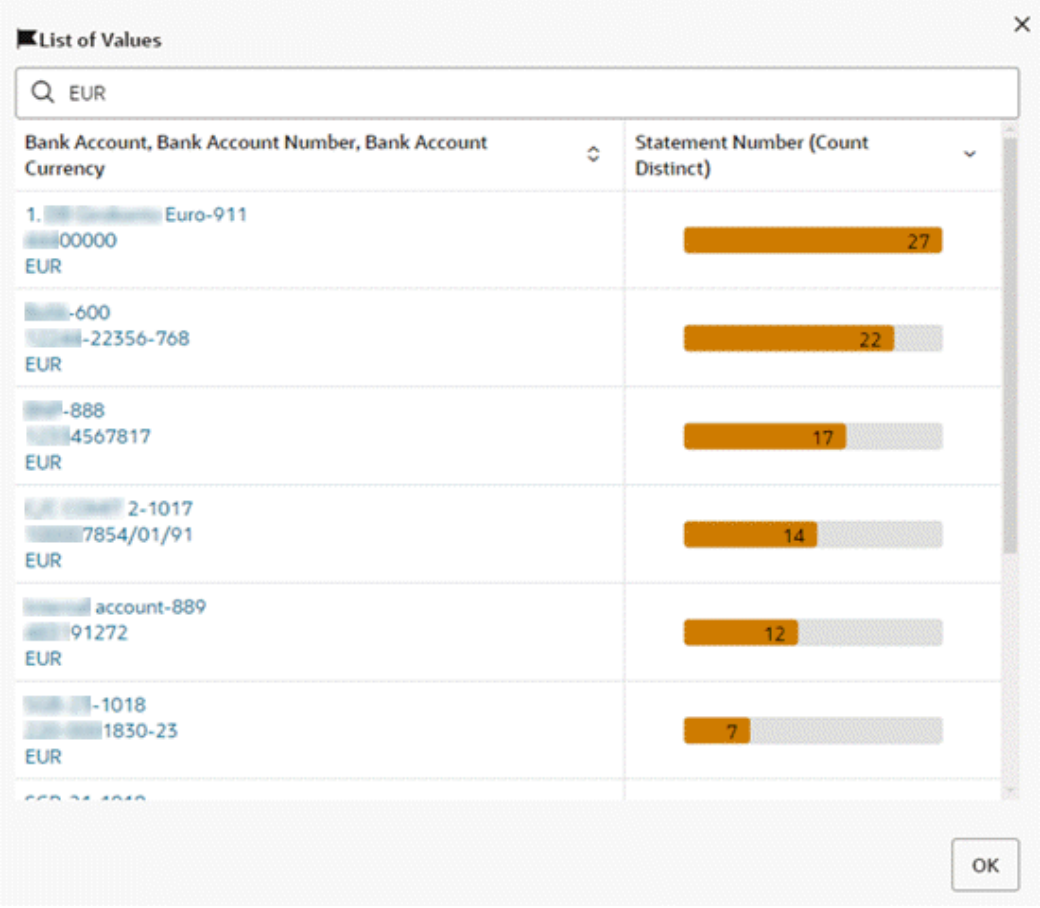
Example of Summarization Bar Personalization Through Runtime Options



The Flag Search feature is introduced in V12. This search capability within a flag pop-up allows users to search within the entire list, rather than being restricted to the 100

items displayed. This search functionality inherits the capabilities available in the Available Refinement feature. It is important to note that this search functionality is applicable exclusively to dimensions.

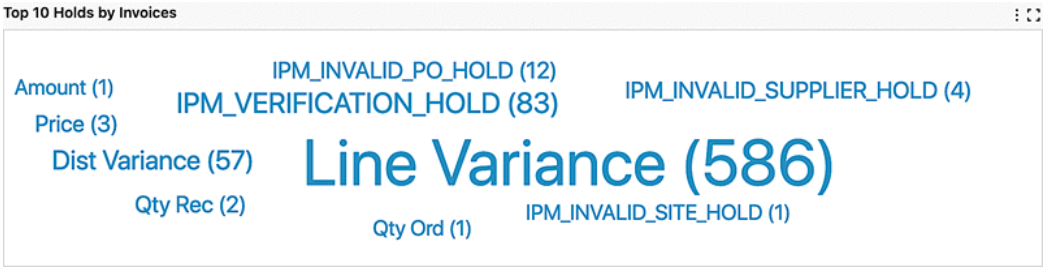
Example of Searching within the Flag Pop-up Window



Tag Cloud

The Tag Cloud component allows users to compare a set of displayed terms based on the value of an associated metric. The component can optionally display the metric value associated with each term.

Tag Cloud Example



Configuration

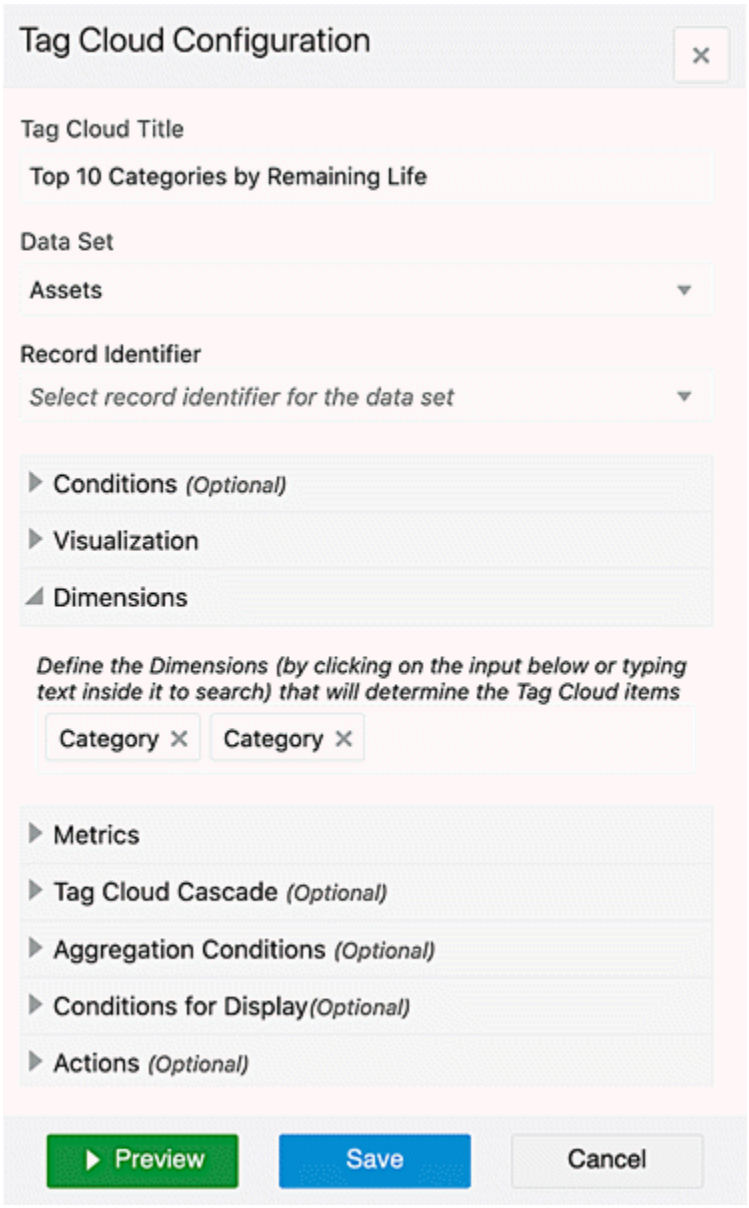
Configuration information for a tag cloud is listed below.

General configuration

The terms displayed on the Tag Cloud are values from a selected dimension. You can configure a list of available dimensions for users to select from.

The value of the selected metric determines the relative size of the Tag Cloud terms.

Tag Cloud Configuration



Specific configuration

Specific configuration options for the Tag Cloud are listed below.

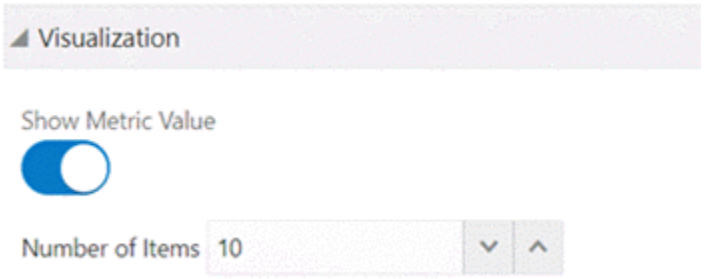
Visualization

Provide additional display configuration options for the Tag Cloud.

Tag Cloud Visualization Options

Option	Description
Show Metric Value	Turn on this option to display the calculated metric value for each Tag Cloud term.
Number of Items	Type or select the maximum number of terms to display on the Tag Cloud.

Tag Cloud Visualization Configuration



Tag Cloud Cascading

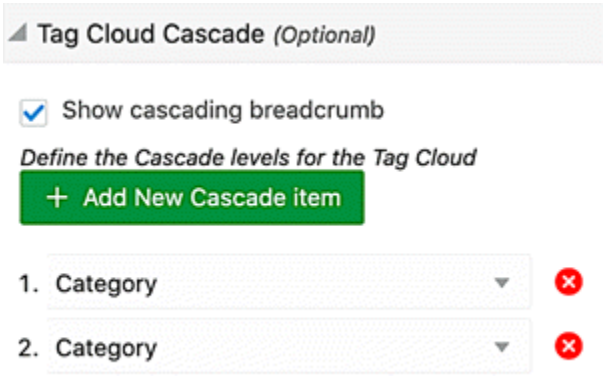
Cascading levels are displayed as a trail of breadcrumbs, called **cascading breadcrumbs** , on the top right corner of tag cloud giving a better understanding of drill-downs applied to reach the current state.

Tag Cloud Cascading Options

Option	Description
Show cascading breadcrumb	Disable to not display cascading breadcrumbs

Option	Description
+ Add Cascading	• Click +Add Cascading Item. • Select the first dimension (the top level of the cascade). • To add another cascade level, click +Add Cascading Item and select the second dimension (the new dimension is added to the end of the cascade). **Note:** To add a cascading level, all dimensions should be redefined in the dimension section.

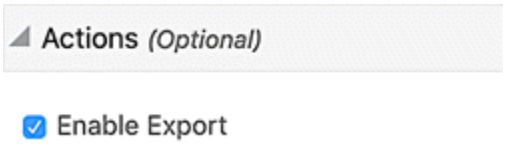
Tag Cloud Cascading Configuration



Export in Tag Cloud

Underlying data of the Tag Cloud component can be exported in a CSV file. Exported data honors all runtime changes. As the underlying data needs to be holistic, exported data of the chart contains all the data irrespective of how many dimensions are displayed. The Export feature is controlled from the Actions accordion.

Tag Cloud Export Configuration



Chart

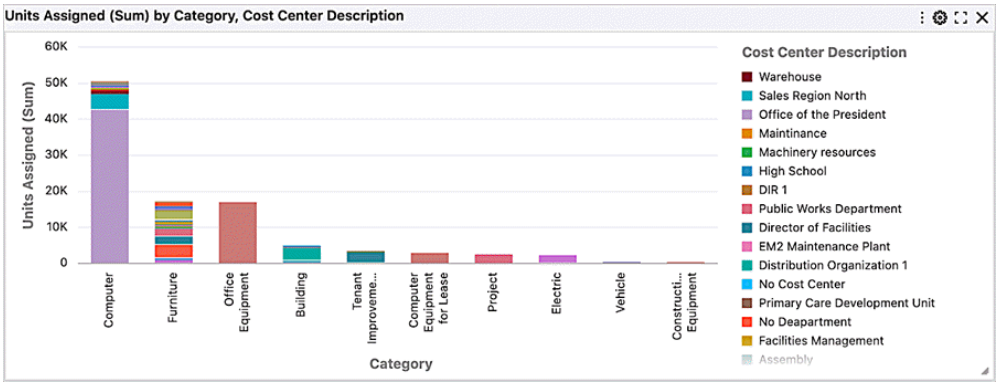
The Chart component displays a graphical chart based on the application data. It supports several sub-types and includes options for selecting the specific data to display.

The chart subtypes are described below.

- Bar Chart

  Bar charts show one or more metric values aggregated across a series dimension or group dimension.

Bar Chart Example

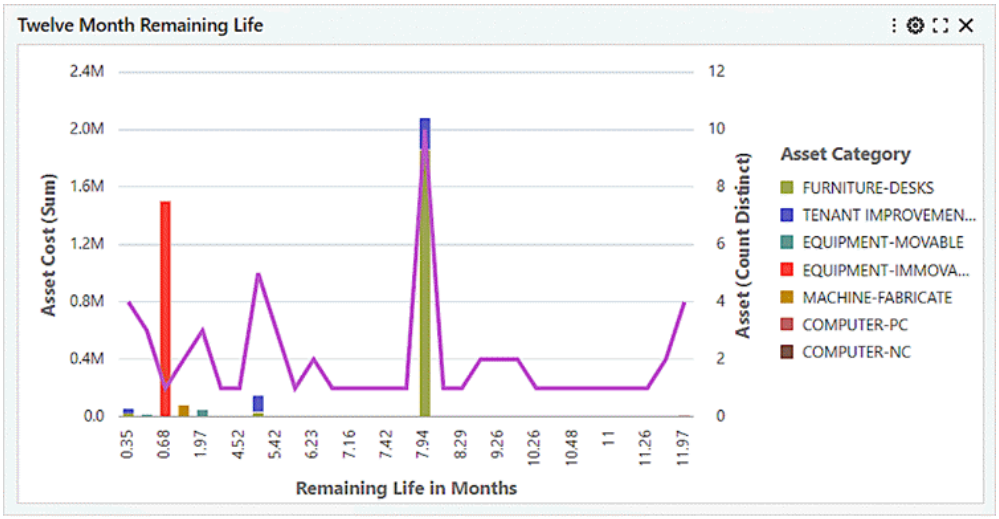


  They are useful for precise comparisons of one or more values.

- Bar/Line Chart

  Bar/Line charts show metric values aggregated across group or series dimensions on two different scales.

Bar/Line Chart Example

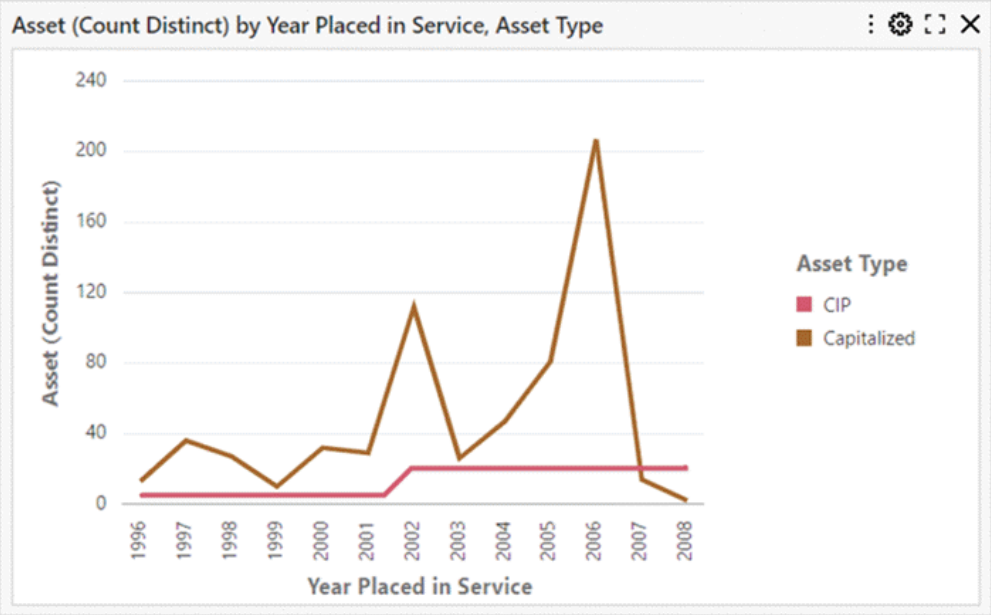


They are useful for showing quantity alongside changes in trends over time.

- Line Chart

  Line charts show one or more metric values aggregated across series dimension or group dimension.

Line Chart Example

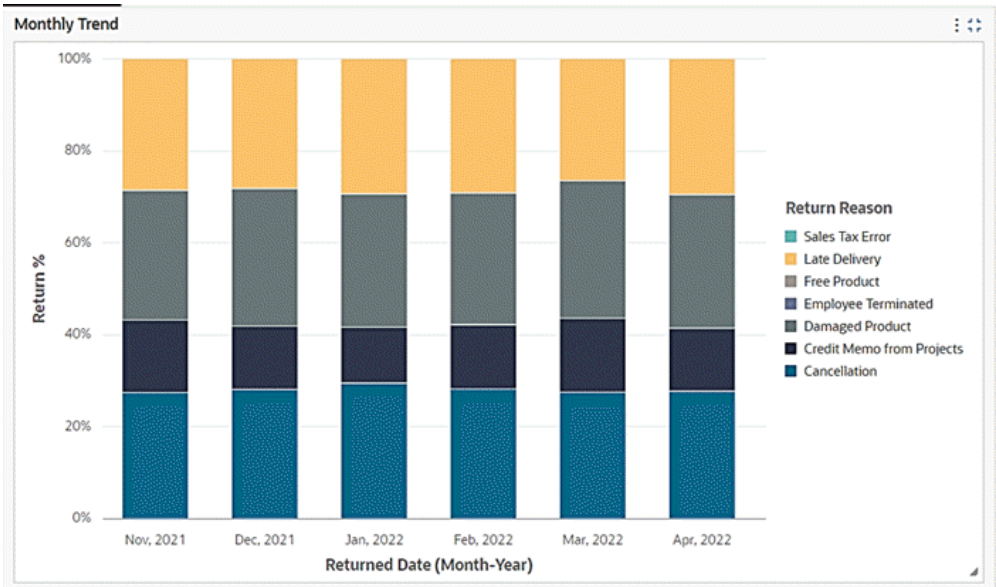


They are useful for showing changes or trends.

- Percentage (Percent) Chart

  Introduced in V10, a percentage chart (also called a percent chart) allows for visualizing the relative percentage of multiple data series in stacked bars, where the total of each stacked bar always equals 100%. Similar to a pie chart, a percentage-stacked bar chart displays the part-to-whole relationship. However, unlike a pie chart, it can also illustrate how proportions change over time. Users have the flexibility to switch the stacked bar chart to a percent chart using runtime options. Additionally, setting the percent chart as the default view is subject to the designer's choice, as it is controlled from the configuration.

Example of a Percentage Chart



A user can change the stacked bar chart to percent chart using runtime options. Setting percent chart as the default view is subject to designer's choice as it is controlled by the configuration.

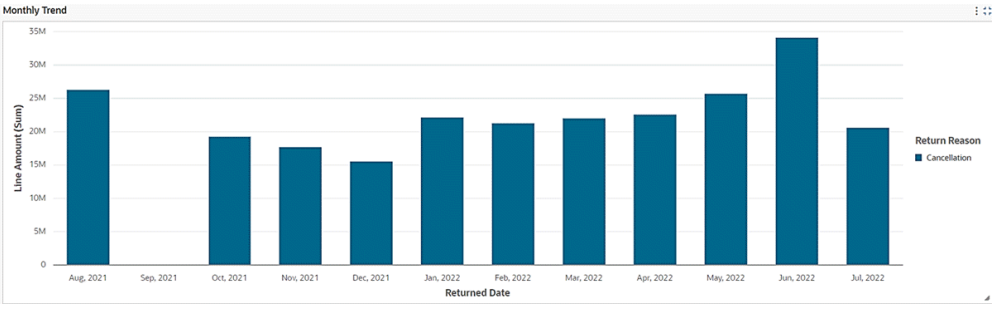
Percent Chart Configuration Options

Option	Description
Show as Percentage	Converts a stacked bar chart to a percent chart, and it is used to set up a percent chart as the default view.

- Time Series Chart

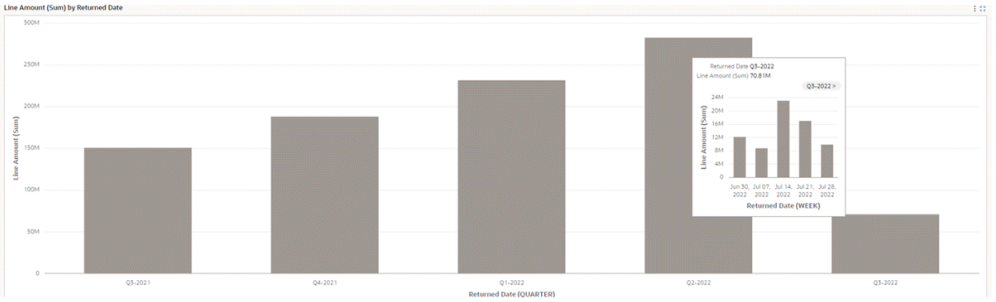
  Introduced in V10, additional configuration options are available for bar and bar/line charts to visualize a series of data points collected over time. This time series chart feature aids in aggregating daily data to monthly, quarterly, or yearly levels, thus facilitating time-based analysis. The time (monthly, quarterly, or yearly) buckets are created dynamically without making any changes to metadata level configurations as in the case of date subset. In V11, the time series chart is enhanced to support finer time grains such as Week, Day, Hour, and Minute. Hour and Minute time grains are only supported during cascading. Cascading is explained in detail under the section on common chart features.

Example of a Time Series Chart with Data on a Monthly Level



Beginning with V11, the time series chart supports finer time grains such as Week, Day, Hour, and Minute. Hour and Minute time grains are only supported during cascading. The cascading drop-down list has auto-bucketed time grain from the start point till the end based on the attribute profile.

Example of a Time Series Chart with Data on Quarterly and Weekly Levels



Time Series Chart Configuration Options

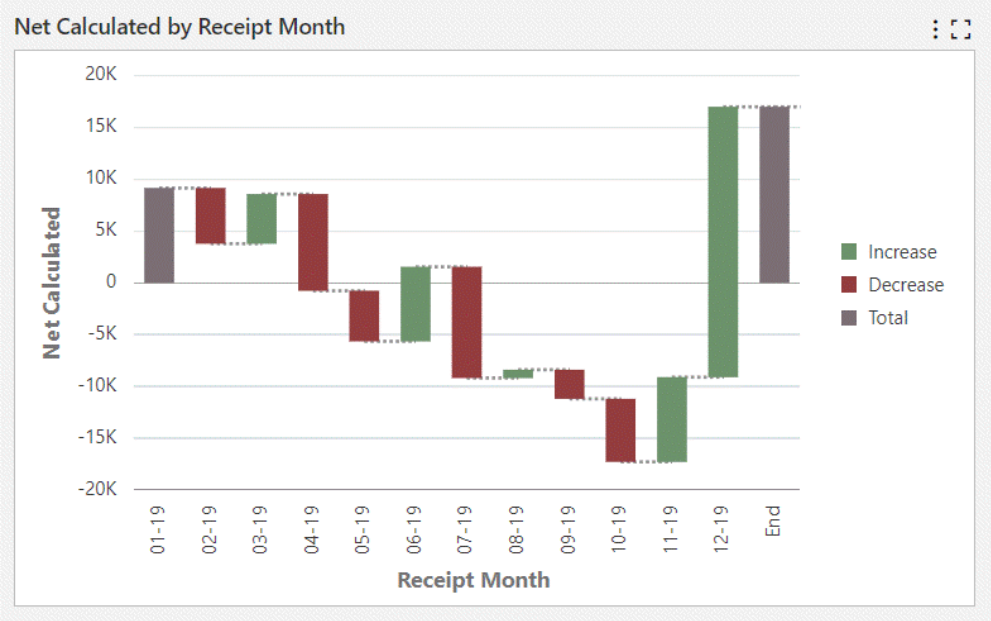
Option	Description
Time Dimension	Used to define any date or date-time attribute as a time dimension. The date or date-time attribute must have been configured as a dimension.
Time Grain	Used to set the aggregation for the time dimension.

- Waterfall Chart

  Introduced in V6, the waterfall chart joins the chart family in ECC, offering a valuable tool to illustrate changes to a value over a period of time or dimension.

They are useful in displaying how the initial state and ending state has changed over a dimension while signifying the major contributors to this change.

Waterfall Chart Example



A waterfall chart can be controlled to display the beginning and ending totals. The value of first dimension is depicted as the beginning total and a dummy "End" bar is added to show the final value.

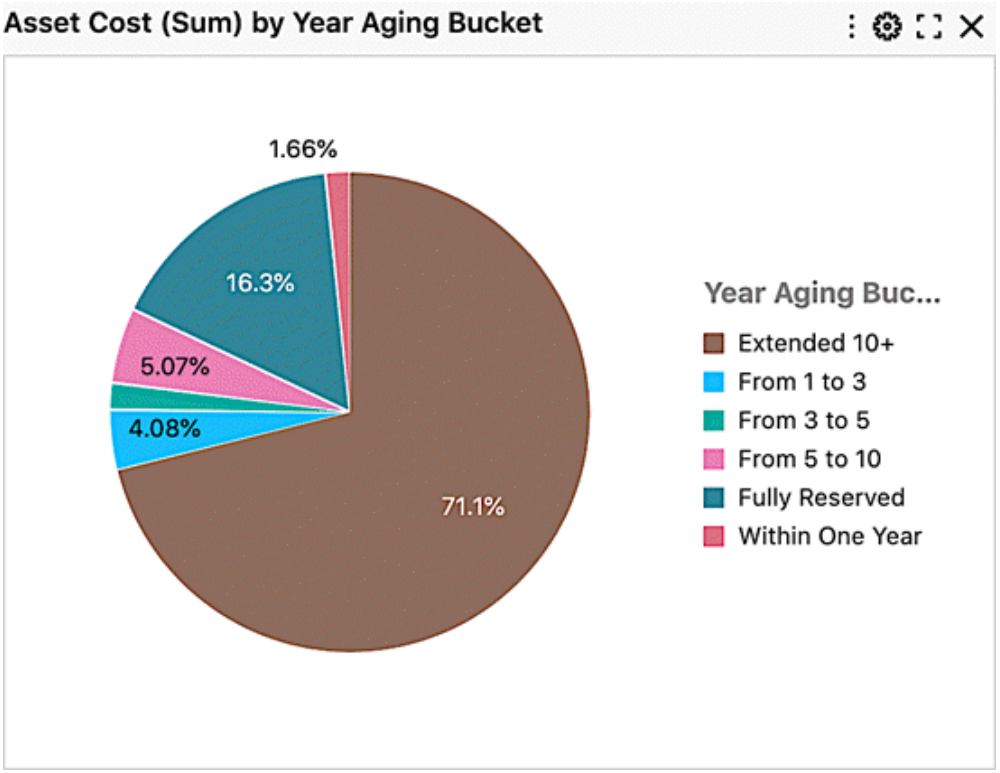
A waterfall chart can also display connecting lines to connect each bars in so that the user understands what is starting value and ending value for every dimension.

The waterfall chart is a subset of the Bar chart and honors the common functionalities of charts.

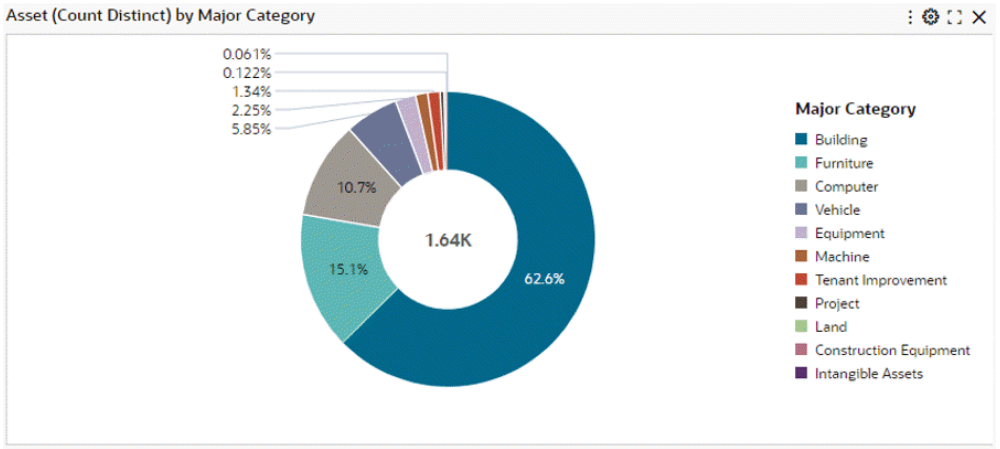
- Pie/Donut Chart

  Pie/Donut charts show a single metric aggregated across a series dimension.

Pie Chart Example



Donut Chart Example



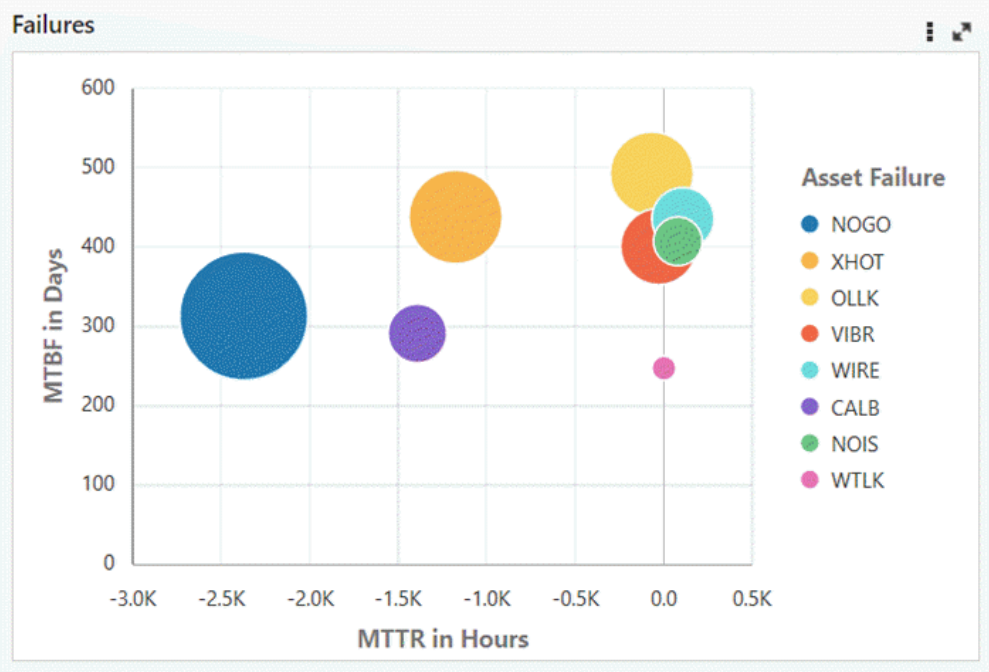
They are useful for showing how each value contributes towards a total.

- Scatter/Bubble Chart

  Scatter/Bubble Chart

Scatter charts display data points, with each point representing a dimension value. Points can also be aggregated into bins in a binned scatterplot, or measured against a third metric and displayed as scaled bubbles in a bubble chart.

Bubble Chart Example

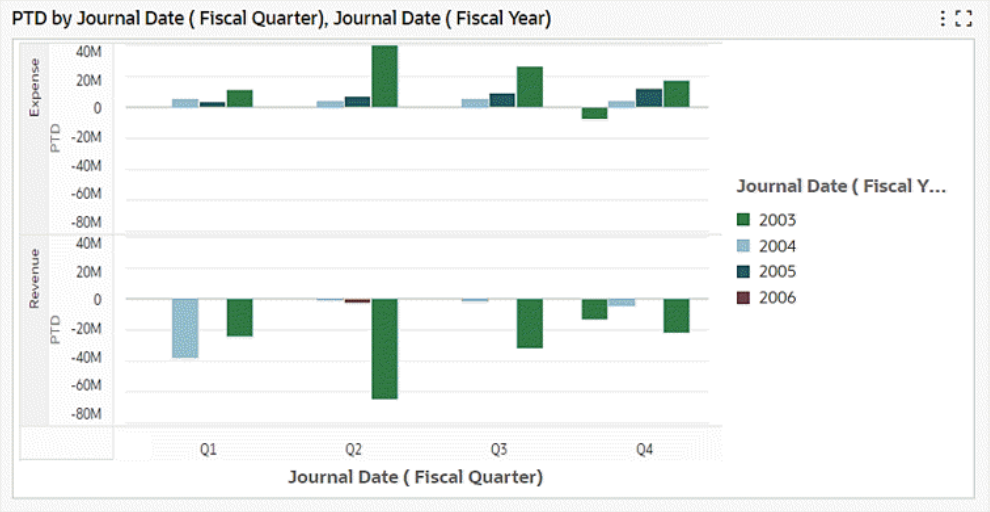


These charts are useful for showing correlations between metrics.

- Trellis Chart

  Introduced in V10, the Trellis Chart displays a series of sub-charts that use the same scale and axes, making relationships among the data easier to understand. A trellis chart splits a chart into multiple versions of itself, presented side-by-side (or one above the other), with its data partitioned across these versions by a chosen series dimension (for example, splitting a "sales by category" column chart across product lines or country). The following example has two versions of a chart for period-to-date balances over journal date, one for expenses and the other for revenue.

Example of a Trellis Chart



Based on the configuration, end users can choose to flip between trellis rows and trellis columns, or choose to remove the Trellis option from the Chart Visualization completely.

Beginning with V11, Pie and Donut charts are also supported in Trellis.

The Trellis Chart runtime options are:

- When user clicks on runtime options, they can see Trellis row/ Trellis column dropdowns if they were configured by the designer respectively. In below example Trellis Row was configured by the designer.

- Trellis rows: If configured by designers, the end user can use runtime options to flip the Trellis into a Row Trellis, example: Add Account type in Trellis rows. End users can then scroll vertically to view all account types with PTD respectively.

- Trellis Columns: If configured by designers, end user can use runtime options to flip the Trellis into a Column Trellis, example: Add Account type in Trellis column. End users can then scroll horizontally to view all account types with PTD respectively.

  A trellis chart can be exported to a CSV file but image export for the trellis chart component is not supported in V10.

  For configuration, when a designer selects the chart component, they are given a Trellis rows and Trellis columns input box in a new Trellis accordion. They can select any dimension as trellis rows or trellis columns.

Multi Data Set Support

Bar and Bar/Line charts support display of metrics from multiple data sets over the common dimensions. The common dimensions should have the same attribute display name and an association should be defined on these data sets.

Association takes care of refining the dashboard appropriately when a filter is applied from the chart. Metrics are aggregated according to the data corresponding to the data set. If a dimension value is missing from any of the data set, it is shown with a corresponding zero value.

Multi data Set Support is applicable for multi-metric charts and honors the existing functionalities of a multi-metric chart.

Once Multi Data Set Support is enabled, a designer can configure the conditions and record identifier for each data set.

Multi Data Set Support also allows users to configure metrics from the same data set but with different conditions or record identifiers.

General Configuration

Configuration options for charts are described below.

General Configuration Options

Option	Description
Title	The title of the chart.
Data Set	Select a data set to associate with the chart.
Enable Multi Data Set	Use this flag to enable configuration support to multiple data sets.
Chart Type	Select a chart type.
Dimension	Series dimension: is mandatory (M) in most of the chart types as described in the table below.
Time Dimension (Optional)	Provision to set a date or date-time attribute as a time dimension and to set its aggregation (Daily, Weekly, Monthly, Quarterly and Yearly).

Chart Dimensions

Type	Group Dimension	Series Dimension
Bar	N/A	(M) Category axis
Stacked Bar	Category axis	(M) Color
Stacked Bar Percentage	Category axis	(M) Color
Multi-Metric Bar	N/A	(M) Category axis
Line	N/A	(M) Category axis
Stacked Line	Category axis	(M) Color
Bar/Line	Category axis	(M) Category axis
Stacked Bar/Line	Category axis	(M) Color
Multi-Metric Bar/Line	N/A	(M) Category axis
Bubble	(M) Color	Number of Bubbles
Scatter	(M) Shape and Color	Number of Shapes
Pie/Donut	N/A	(M) Wedge color

In the Chart Dimensions window, you can define the dimensions for a chart. Click in the search box to see a list of dimensions, or enter text to search for a specific dimension.

Use the Time Dimension configuration window to optionally define the options for the time dimension. For example, the Time Dimension could be 'Returned Date' and the Time Grain could be "Monthly'.

Example of a Chart with a Reference Line

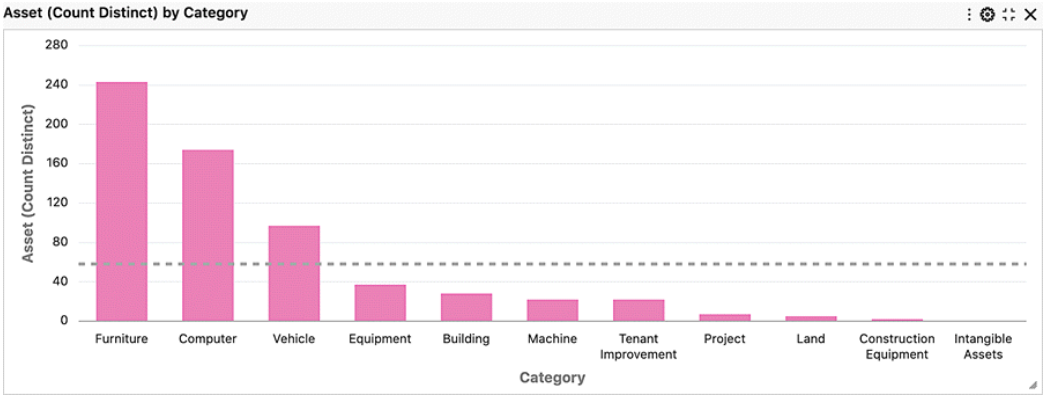
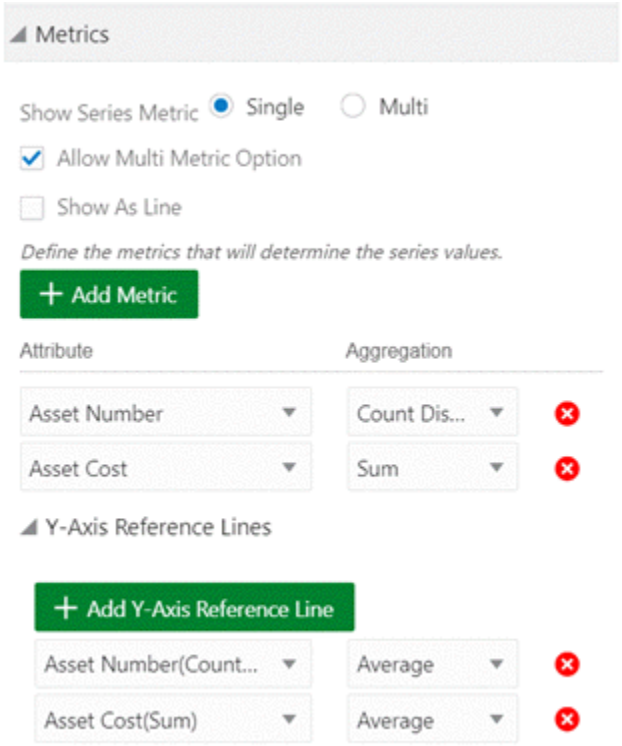


Chart Options

Option	Description
Metric	
Show Series Metric (Single vs. Multi)	Controls the display of metric values all at once on a chart, or controlled by the user through chart runtime options.
Allow Multi Metric Option	Allow business users at runtime to change the chart from single to multi- metric. This option is checked by default.
Show as Line	Convert a bar chart to a line chart.
Show as Waterfall	Convert a bar chart to a waterfall chart. The waterfall chart component supports only a bar chart with a single dimension and a single metric.

Option	Description
+ Add Metric	Select metric attribute + aggregation function as below: • SUM: Calculates the total value for the metric. • AVERAGE: Calculates the average value for the metric. • MINIMUM: Selects the minimum value for the metric. • MAXIMUM: Selects the maximum value for the metric. • COUNT DISTINCT: Counts unique values for an attribute
Return Absolute Value	Convert negative values to absolute values.
+ Add Y Axis Reference Line	• Reference line: allow displaying reference line on the chart based on a predefined metric. Example: Average of Asset Cost (Sum) • Can add a reference line for as many as the metrics defined on the chart. • If the chart displays only one metric, the corresponding reference line to this metric will be displayed. • Supports average, min, max, constant. • Applied to all chart types except pie/donut charts.

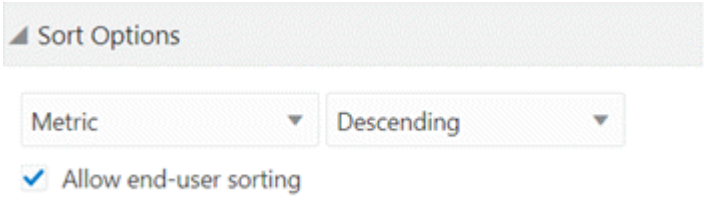
Chart Metric Calculation



Sort Options:

- Control sort order and sort attribute. Either the dimension or the metric can be specified as the sort key. Alphabetical sorting applies to dimensions while numerical applies to metrics.
- Allows end user sorting; enables the end user to change the default sorting at runtime

Chart Sort Option

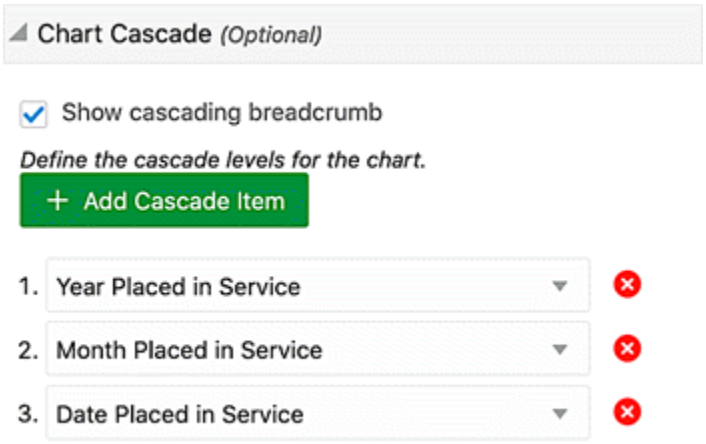


Cascading levels are displayed as a trail of breadcrumbs, called **cascading breadcrumbs** , on the top right corner of a chart. This feature gives the user a better understanding of drill-downs applied to reach the current state.

Chart Cascading Options

Option	Description
Show cascading breadcrumb	Disable to not display cascading breadcrumbs.
+ Add Cascading Item	Select the first dimension (the top level of the cascade). To add another cascade level, click +Add Cascading Item and select second dimension (The new dimension is added to the end of the cascade). **Note:** To add a cascading level; all dimensions should be redefined in the series dimension section.

Chart Cascading Configuration



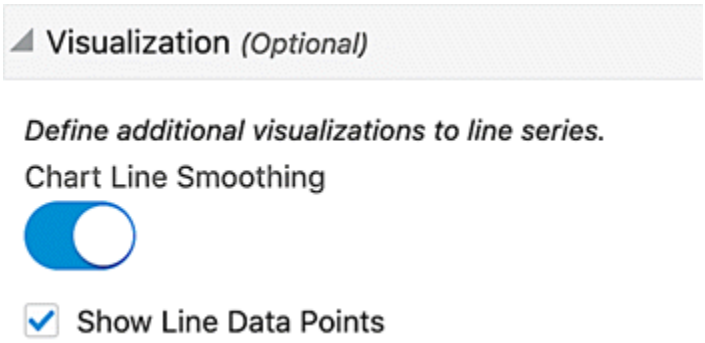
Additional Options for Line and Bar/Line Charts

This section describes configuration options for line charts. For more information on line charts, refer to Data Visualization, *Oracle E-Business Suite User's Guide*.

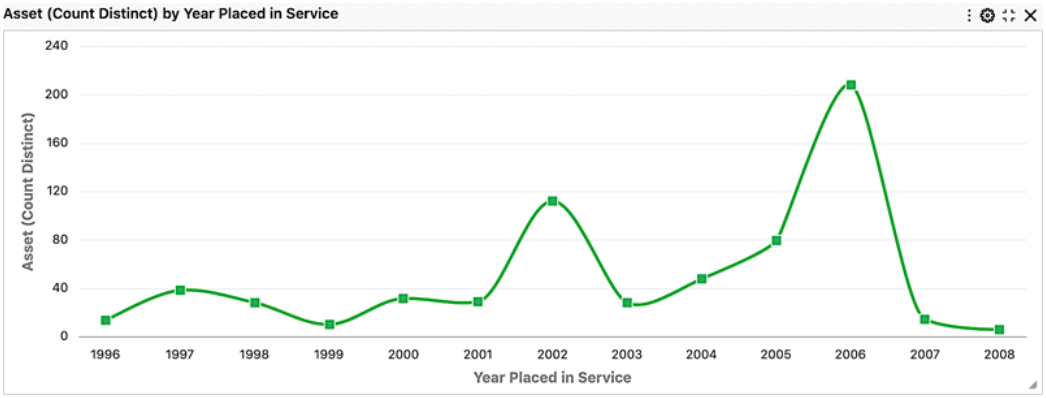
Chart Visualization Options

Option	Description
Chart Line Smoothing	Enable the flag to display a smooth line with curvature. Applicable for line and bar/line charts.
Show Line Data Points	Controls the display of data points on the line. Applicable for line and bar/line charts.

Visualization Configuration for Line and Bar/Line Chart Options



Example of a Line Chart with Smoothing and Data Points

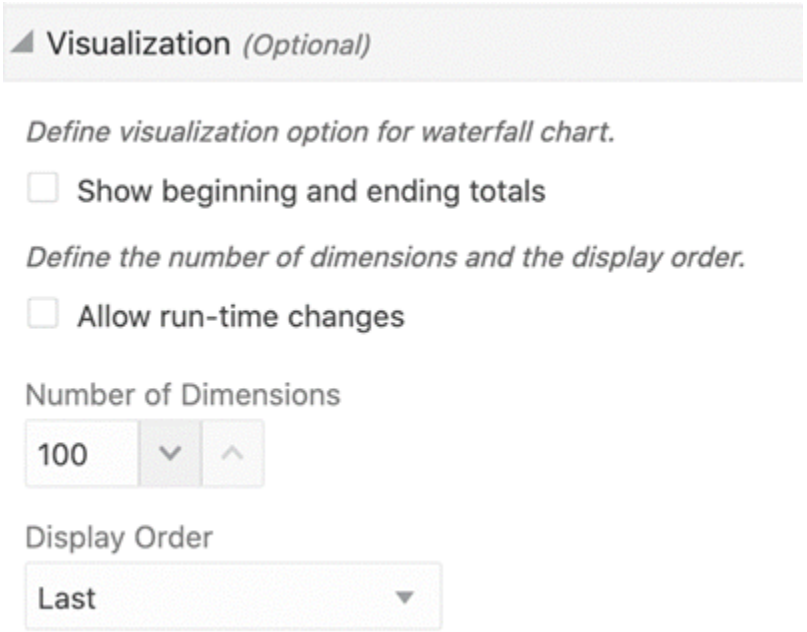


Additional Options for Waterfall Charts

Configuration Options for Waterfall Charts

Option	Description
Show as Waterfall	Converts bar chart to waterfall chart. Note: Group dimension should not be configured as waterfall chart does not honor stacking.
Show beginning and ending totals	Displays totals in the chart. Also enables the runtime option allowing the user to control the display of total bars.
Color Pinning	Controls pinning colors for the increase bar, decrease bar, and total bar. Note: Color pinning in configuration is enabled by default.

Visualization Configuration for Waterfall Chart



"Other" in Pie/Donut Charts

Dimensions corresponding to a percentage lower than the threshold are grouped into the "Other" group. If configured, the threshold limit is also controlled from runtime options.

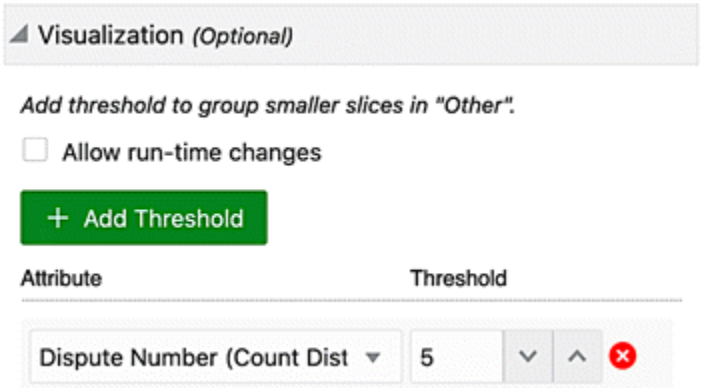
"Other" is always displayed last in the legend. "Other" considers sorting based on the metric – when sorted in ascending order, "Other" is the first group and when sorted in descending order, "Other" is the last group.

"Other" is always displayed at the end of the legend regardless of the sorting option on the dimension. A user can filter by "Other" from the chart to drill down to data grouped in "Other". Also, the user cannot filter by "Other" from the chart legend.

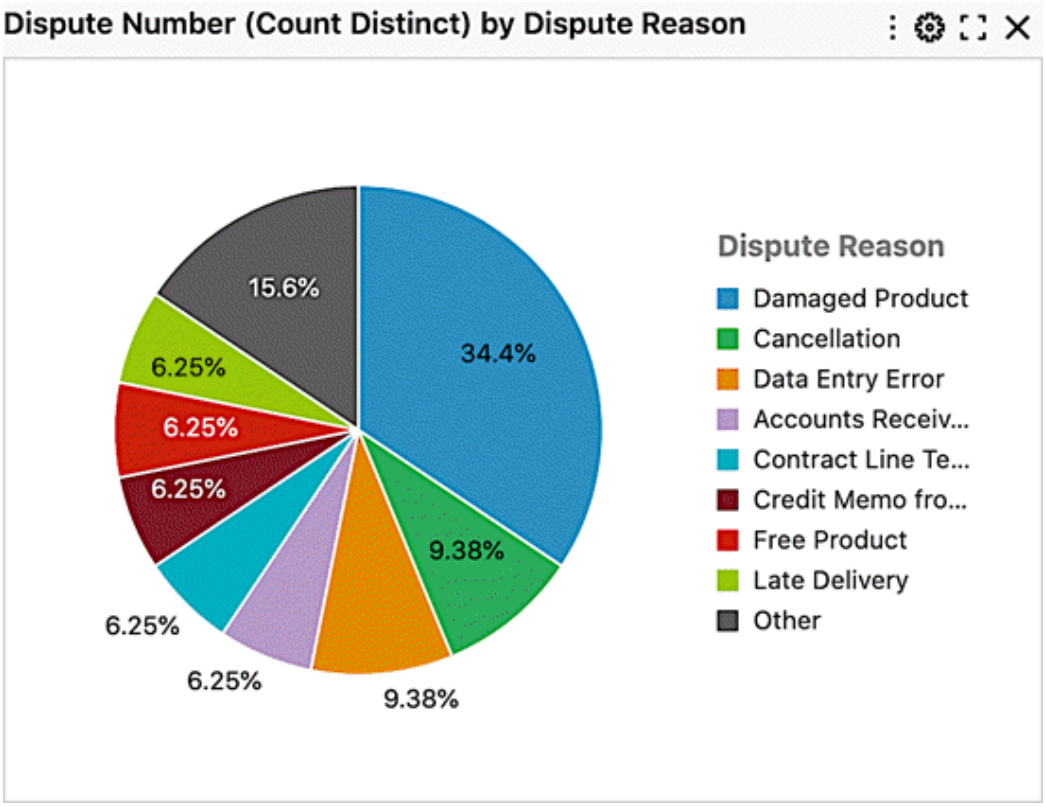
Options for "Other" in Pie/Donut Charts

Option	Description
Allow run-time changes	Enable the flag to allow the business user to make changes to the threshold.
+ Add Threshold	Select each metric and configure threshold corresponding to that metric. The dimensions with metric values below this threshold will be grouped into "Other". Click **+ Add Threshold** for adding a threshold to another metric.

Visualization Options for "Other" Feature in Pie/Donut Charts



Example of "Other" in a Pie Chart



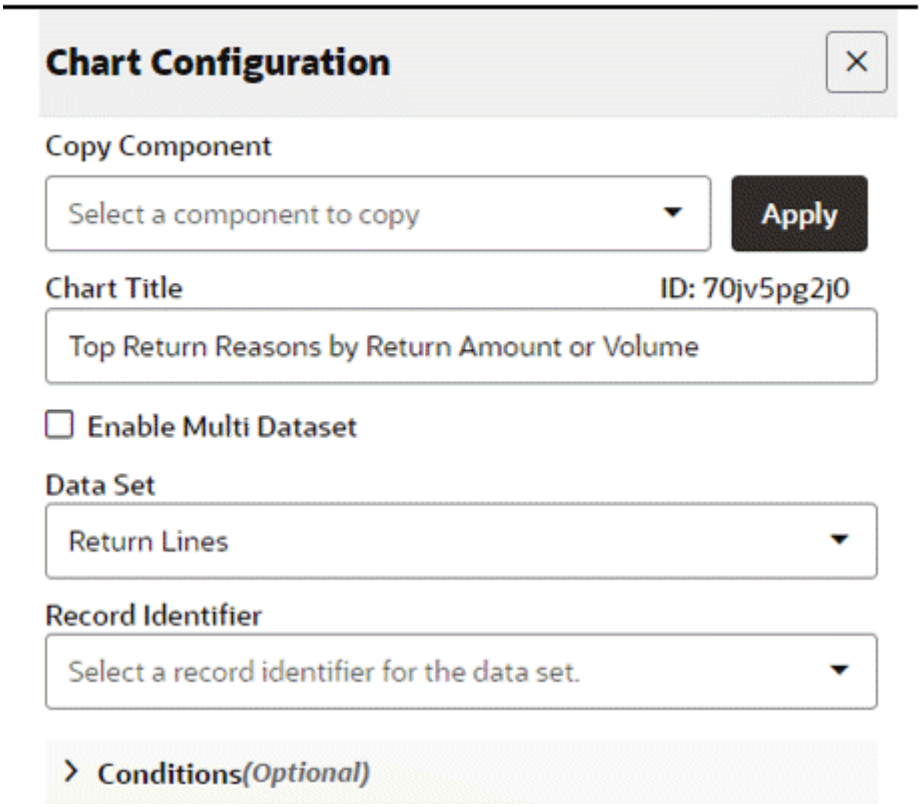
Examples of Configuration Steps for Charts

Bar Chart:

1. Define the chart title and data set.

   Note: The chart title is an optional parameter. Unless it is explicitly set, Enterprise Command Center Framework generates a title based on the dimensions and metrics used in the chart.

Configuration for Chart Title and Data Set



2. Define the chart type.

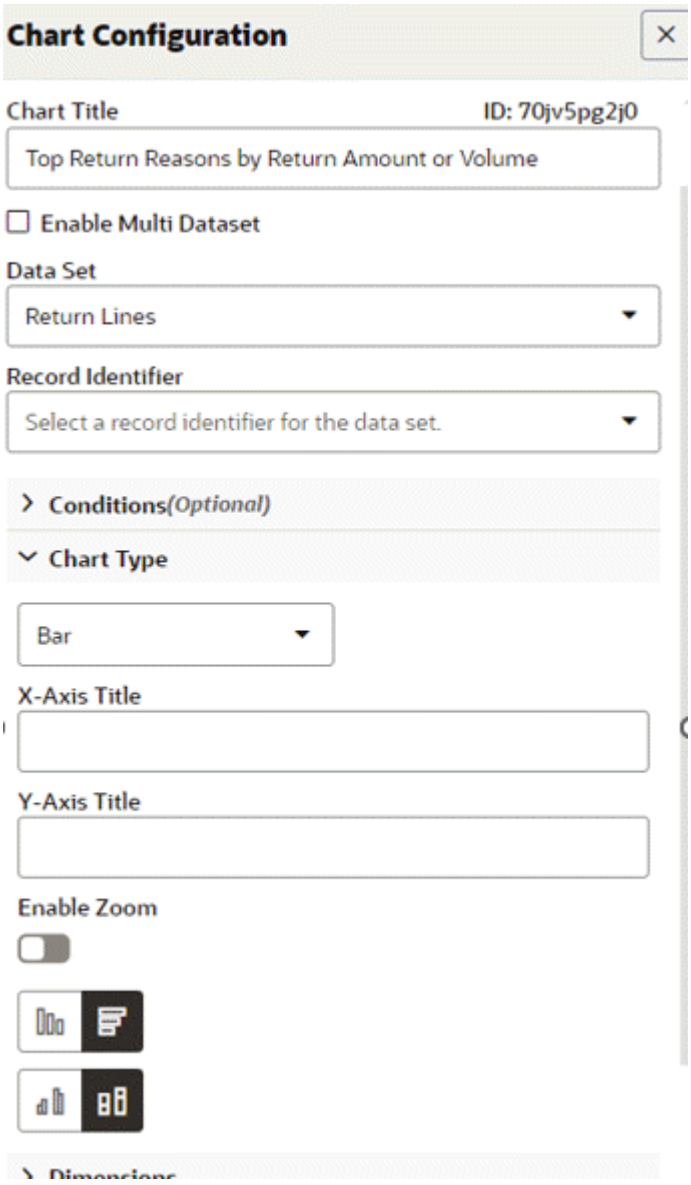
   The x-axis and y-axis titles are optional as ECC can generate them from the dimensions and metric selected.

   Enable zoom if required. Enabling zoom allows the user to zoom in at runtime. Zooming helps enhance the readability of the chart when the metric values are small.

   Define the layout:

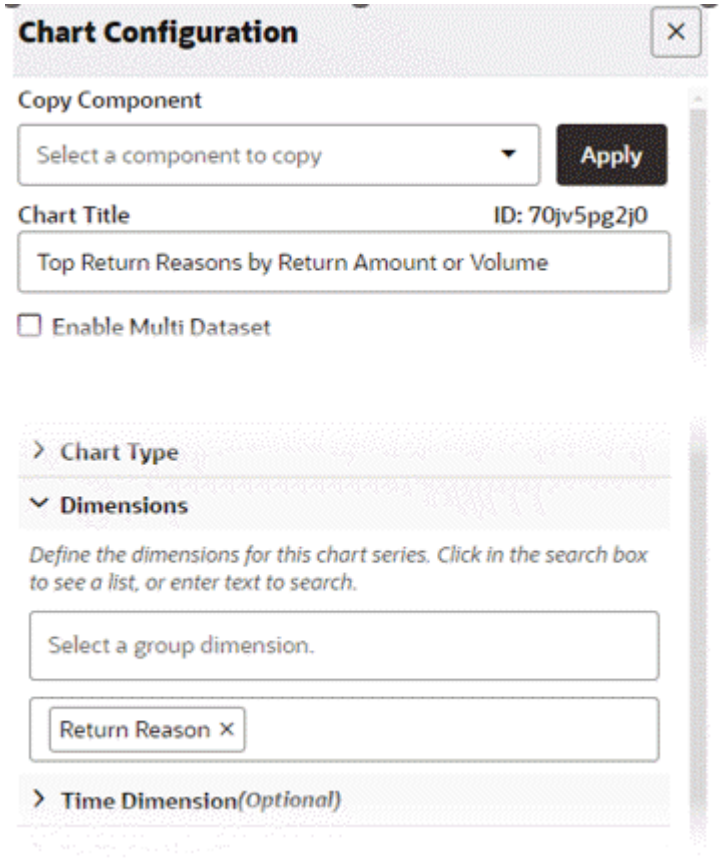
   - Horizontal versus Vertical
   - Stacked versus Unstacked (applicable to a stacked chart only)

Configuration of the Chart Type



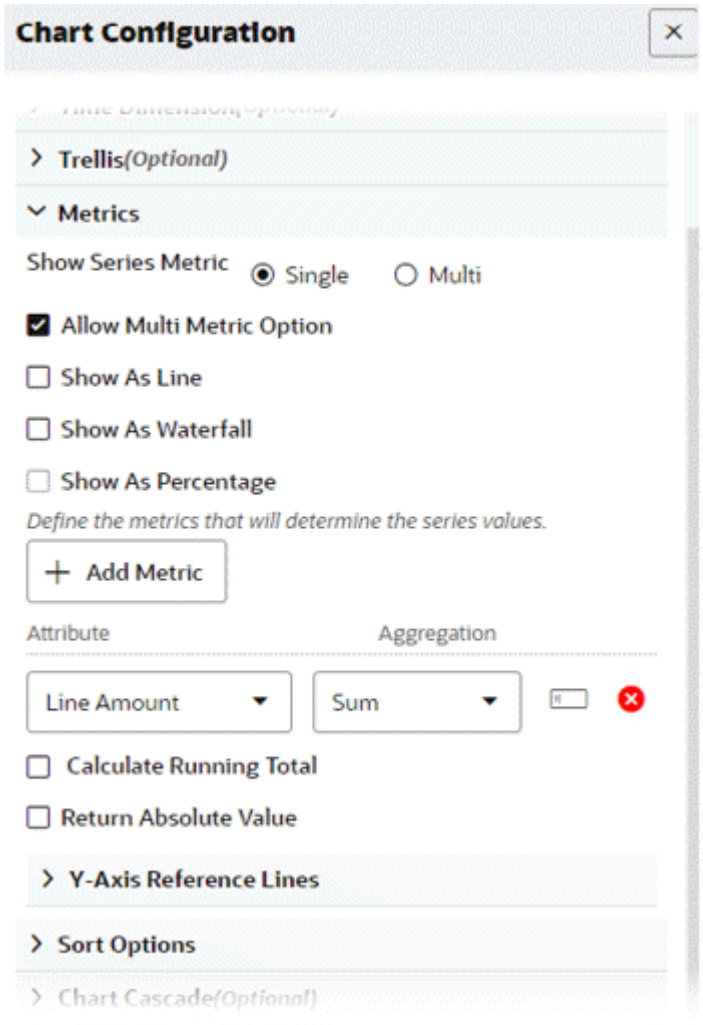
3. Set the series dimension.

Configuration of the Series Dimension



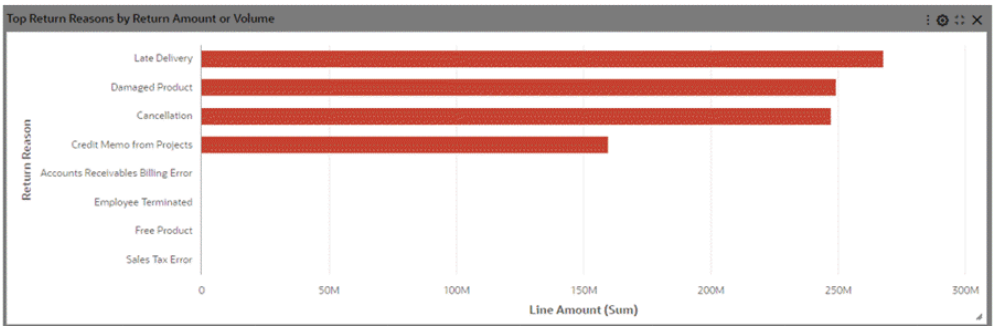
4. Set the metric.

Chart Configuration of Metrics



5. Preview the chart.

Preview of a Bar Chart

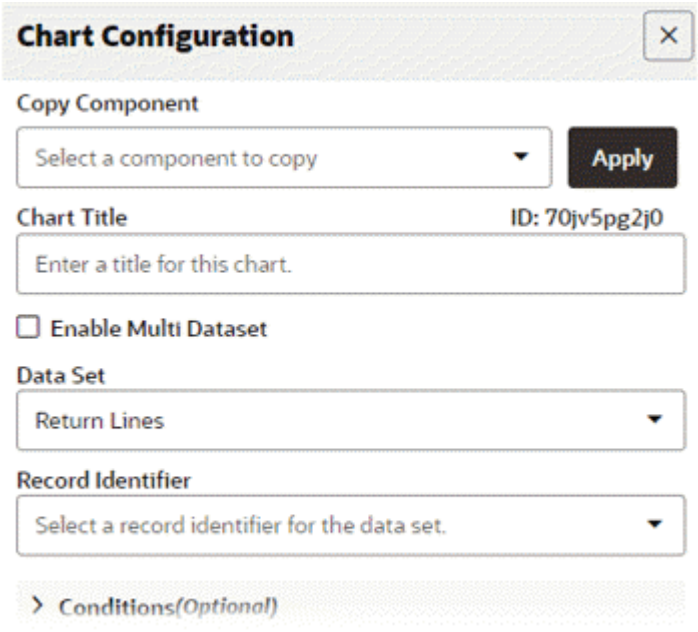


Stacked Bar Chart:

1. Define the chart title and data set.

   Note: The chart title is an optional parameter. Unless it is explicitly set, Enterprise Command Center Framework generates a title based on the dimensions and metrics used in the chart.

Configuration of the Chart Title and Data Set



2. Define the chart type.

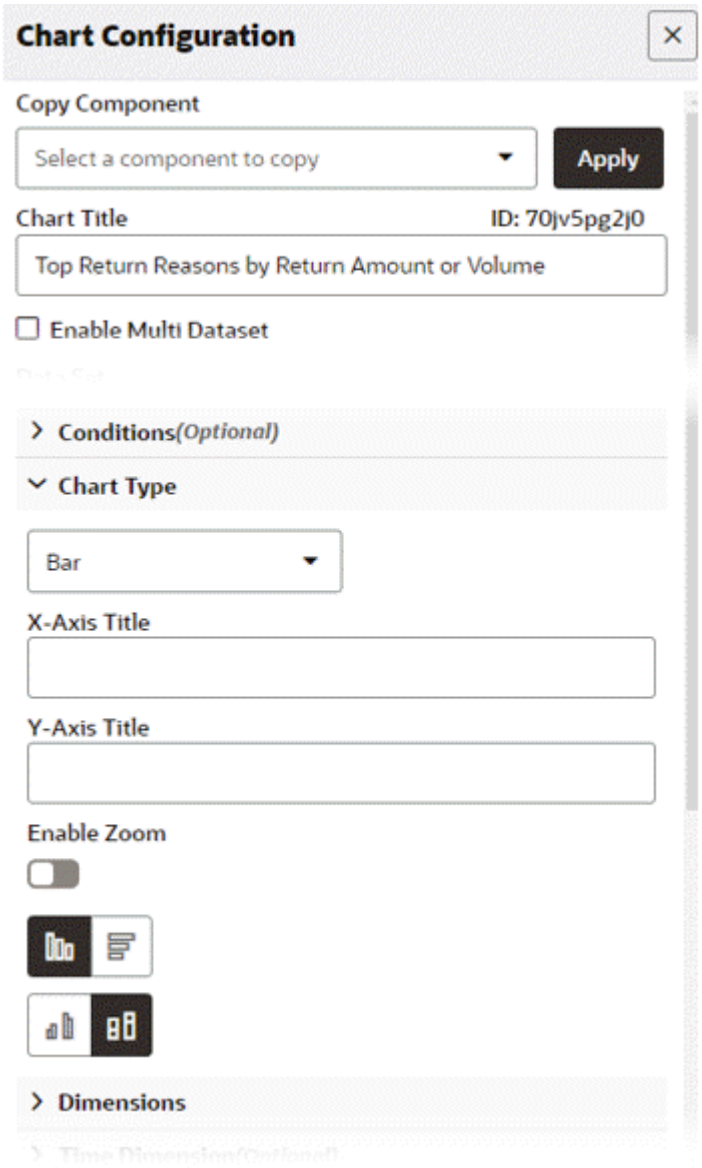
The x-axis and y-axis titles are optional as ECC can generate them from the dimensions and metric selected.

Enable zoom if required. Enabling zoom allows the user to zoom in at runtime. Zooming helps enhance the readability of the chart when the metric values are small.

Define the layout:

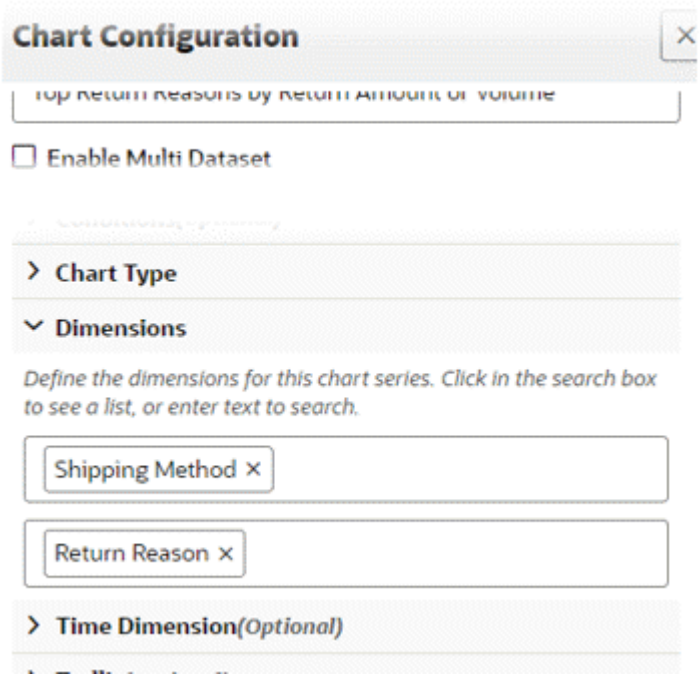
- Horizontal versus Vertical
- Stacked versus Unstacked (applicable to a stacked chart only)

Configuration of the Chart Type



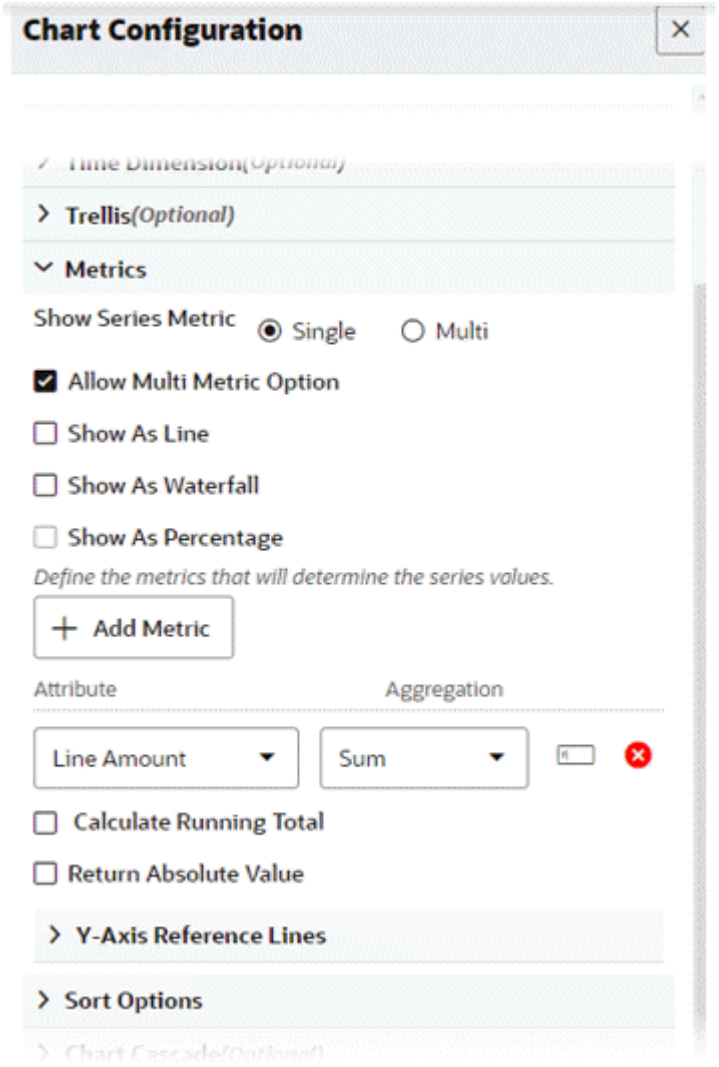
3. Set the group and series dimension.

Chart Configuration of the Group and Series Dimensions



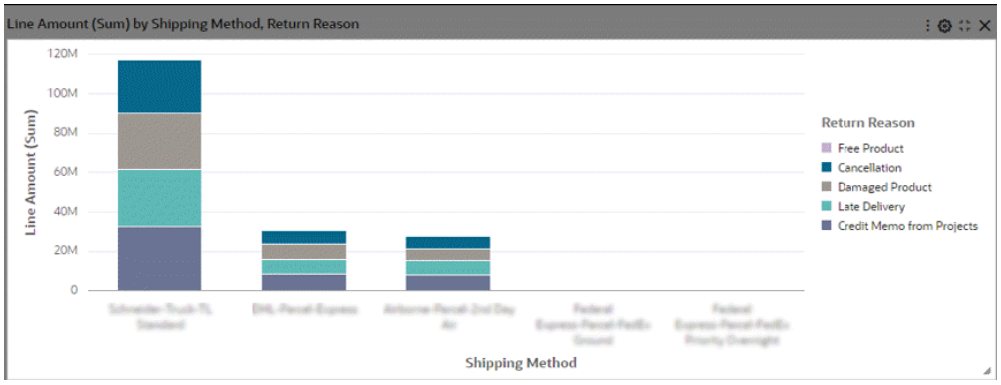
4. Set the metric.

Chart Configuration of Metrics



5. Preview the chart.

Preview of a Stacked Bar Chart

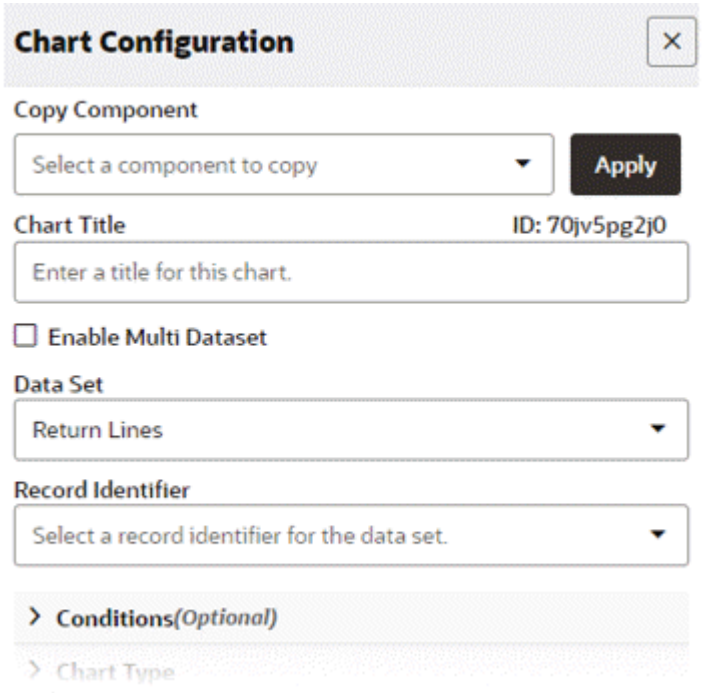


Bar/Line Chart:

1. Define the chart title and data set.

> **Note:** The chart title is an optional parameter. Unless it is explicitly set, Enterprise Command Center Framework generates a title based on the dimensions and metrics used in the chart.

Configuration of the Chart Title and Data Set



2. Define the chart type.

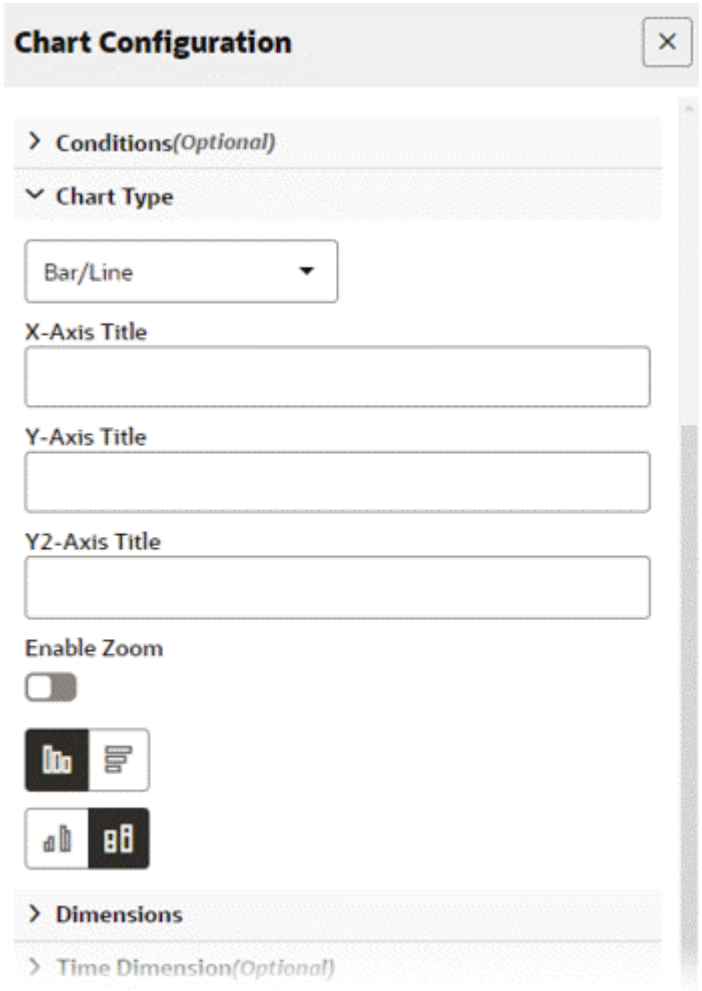
   The x-axis and y-axis titles are optional as ECC can generate them from the dimensions and metric selected.

   Enable zoom if required. Enabling zoom allows the user to zoom in at runtime. Zooming helps enhance the readability of the chart when the metric values are small.

   Define the layout:

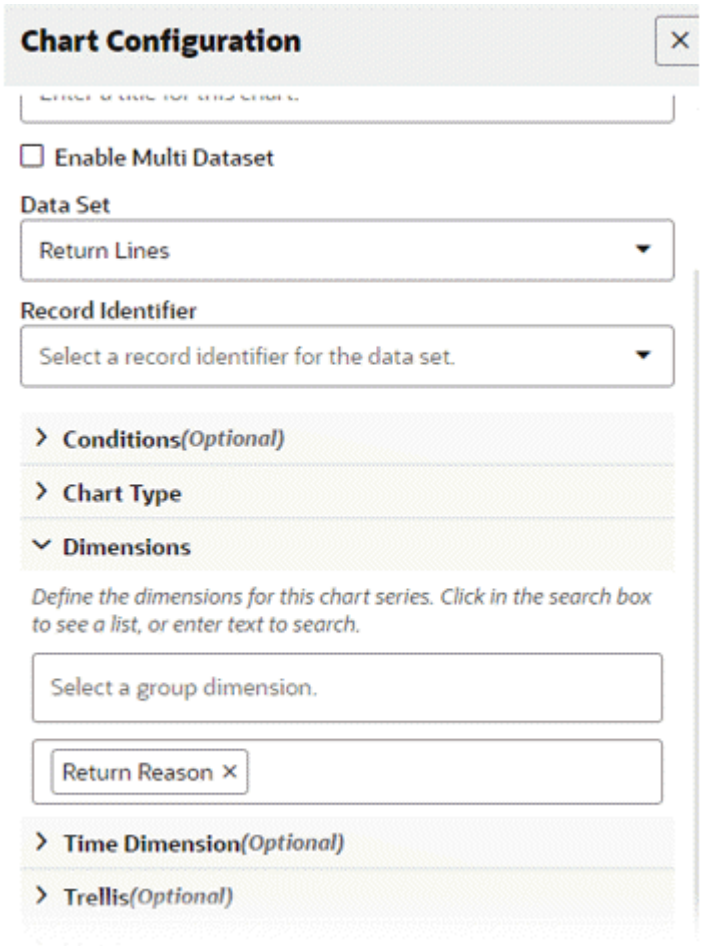
   - Horizontal versus Vertical
   - Stacked versus Unstacked (applicable to a stacked chart only)

   Configuration of the Chart Type



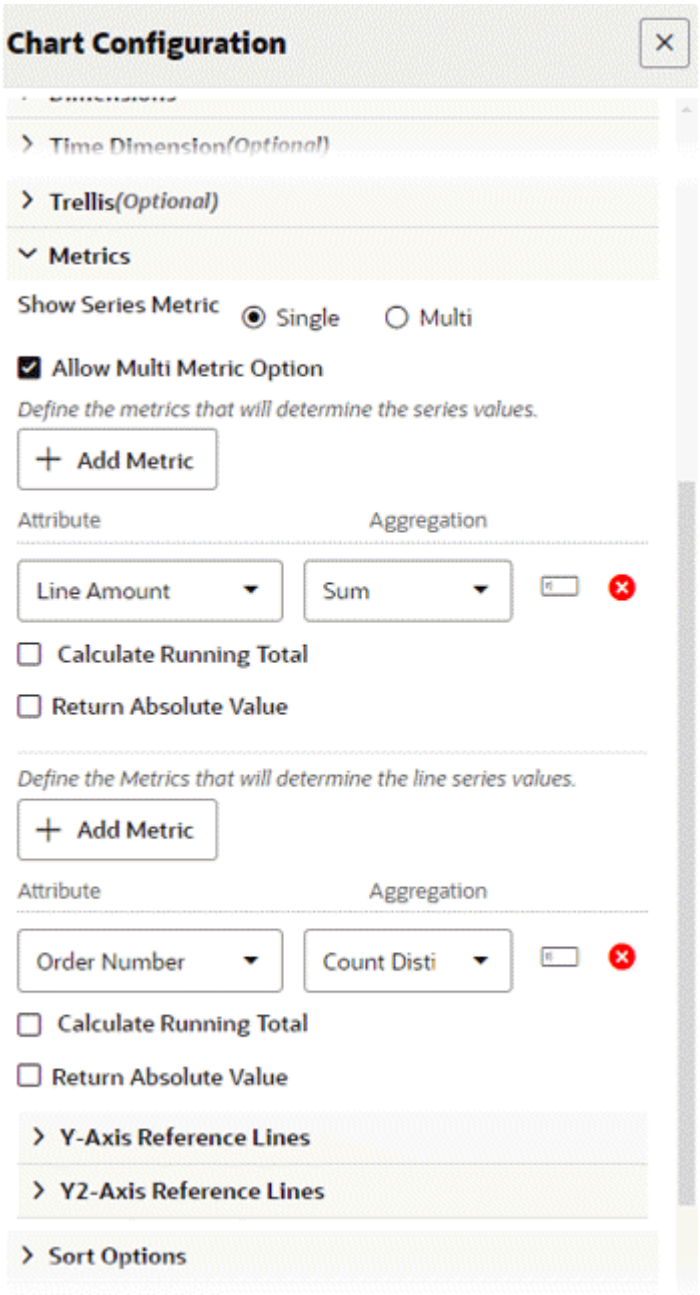
3. Set the series dimension.

Configuration of the Series Dimension



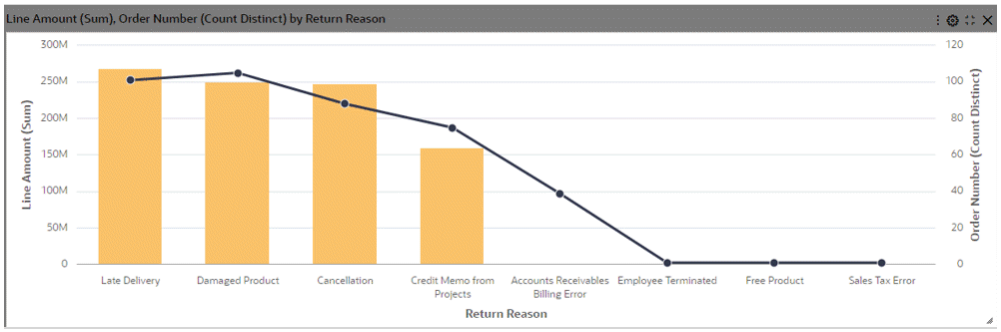
4. Set the metric for the bar and the metric for the line.

Configuration of the Metrics for the Bar and the Line



5. Preview the chart.

Preview of a Bar/Line Chart



Single Metric versus Multi-Metric View:

1. Define the chart title and data set.

   Note: The chart title is an optional parameter. Unless it is explicitly set, Enterprise Command Center Framework generates a title based on the dimensions and metrics used in the chart.

2. Define the chart type.

   The x-axis and y-axis titles are optional as ECC can generate them from the dimensions and metric selected.

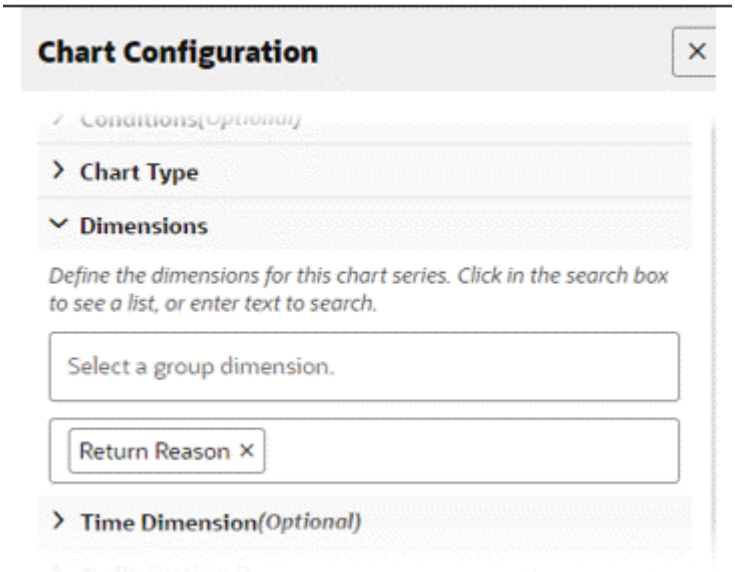
   Enable zoom if required. Enabling zoom allows the user to zoom in at runtime. Zooming helps enhance the readability of the chart when the metric values are small.

   Define the layout:

   - Horizontal versus Vertical
   - Stacked versus Unstacked (applicable to a stacked chart only)

3. Set the series dimension.

Configuration of the Series Dimension

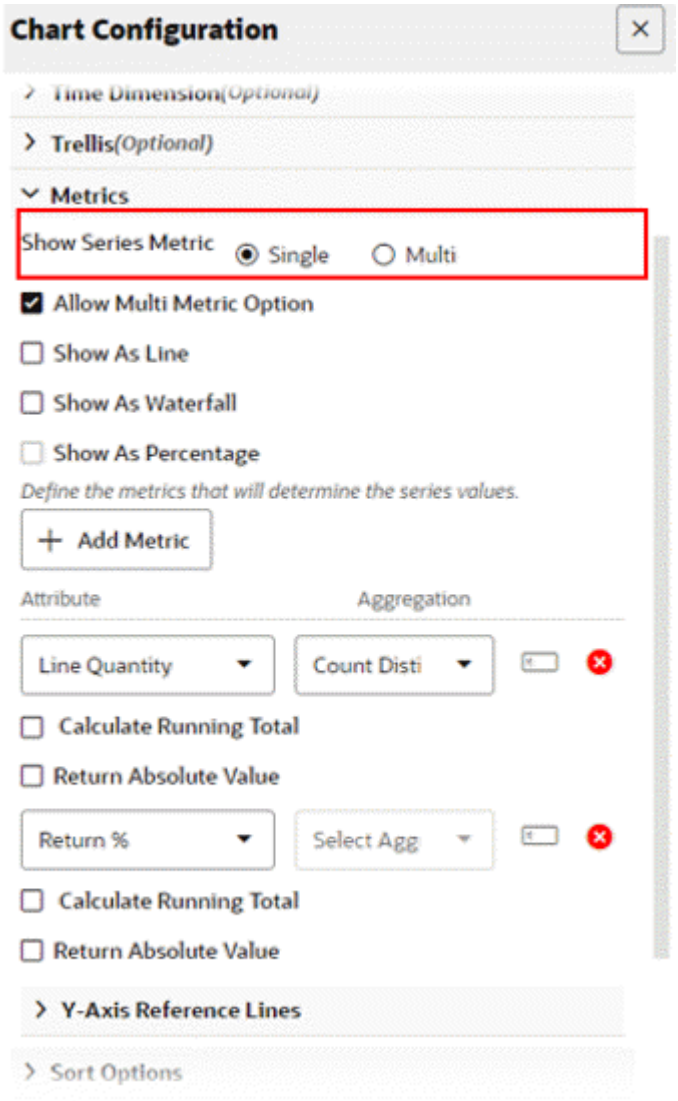


4. Set multiple metrics.

   Note here that for "Return %", no aggregation has been defined. It is a calculated attribute, and hence the aggregation is already defined for it in metadata.

   Toggle between values of "Show Series Metric: Single Vs Multi."

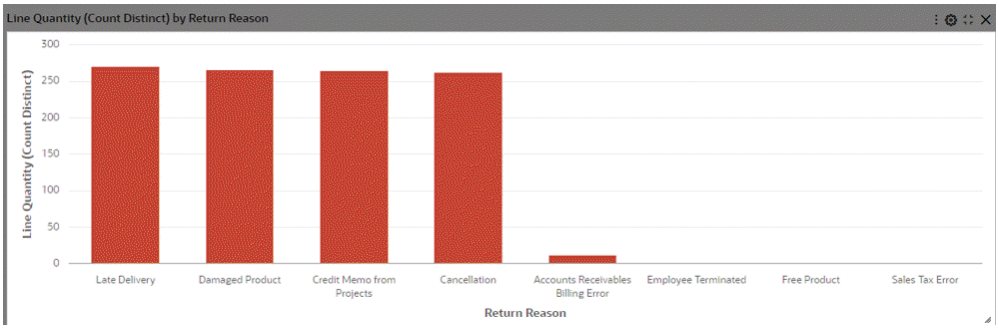
Configuration of Multiple Metrics



5. Preview the chart.

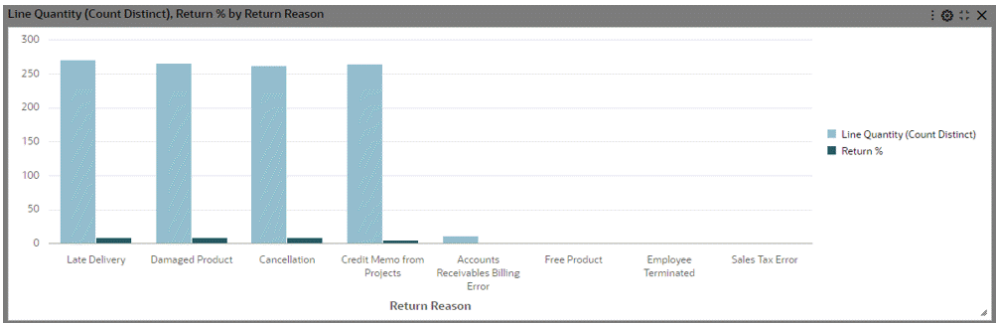
   When the **Show Series Metric** value is set to Single, the chart displays values for that single metric.

Preview of a Bar Chart for a Single Metric



When the **Show Series Metric** value is set to Multi, the chart displays data for multiple metrics.

Preview of a Bar Chart for Multiple Metrics



Multi-Data Set Support in Bar and Bar/Line Charts:

Bar and Bar/Line charts support the display of metrics from multiple data sets over the common dimensions. The common dimensions should have the same attribute display name, and an association should be defined on these data sets.

The association takes care of refining the dashboard appropriately when a filter is applied from the chart. Metrics are aggregated according to the data corresponding to the data set.

If a dimension value is missing from any of the data sets, it is shown with a corresponding zero value.

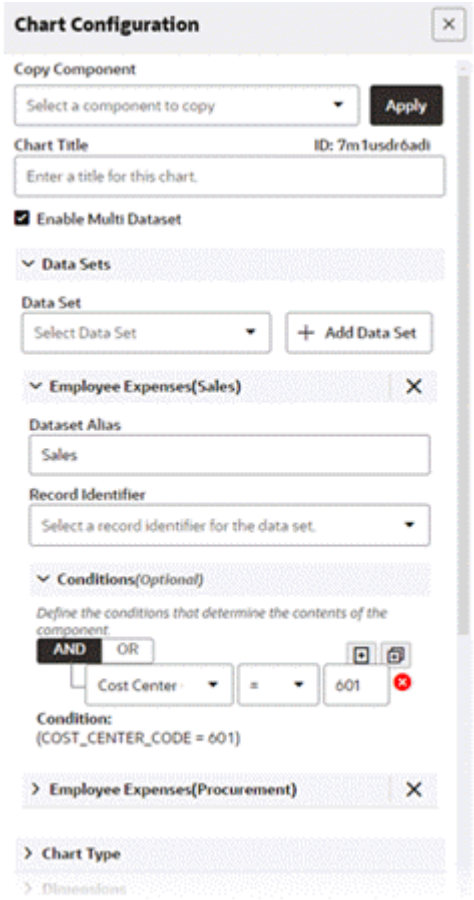
Multi Data Set support is applicable for multi-metric charts and honors the existing functionalities of multi-metric chart. Once Multi Data Set support is enabled, the designer can configure conditions and record identifier for each data set. Multi Data Set support also allows users to configure metrics from the same data set but with different conditions or record identifiers.

1. Define a Data Set 1 with a condition.

Define a meaningful alias for the data set.

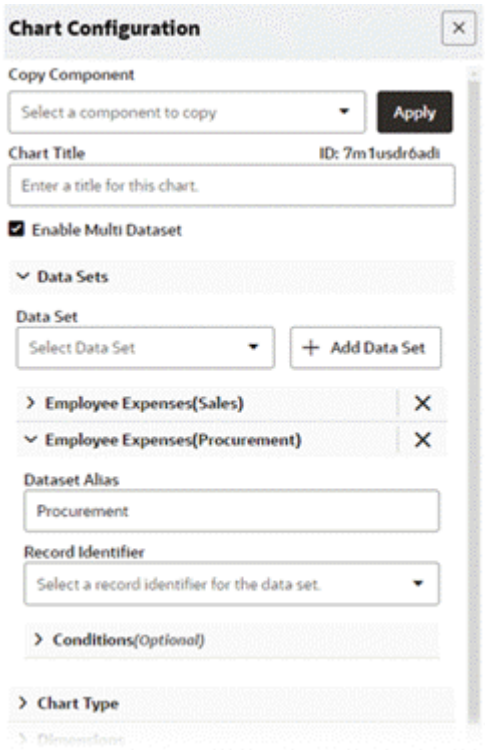
> **Note:** In this example, we are trying to compare expense across categories for Sales and procurement cost centers.

Define First Data Set for Chart Configuration



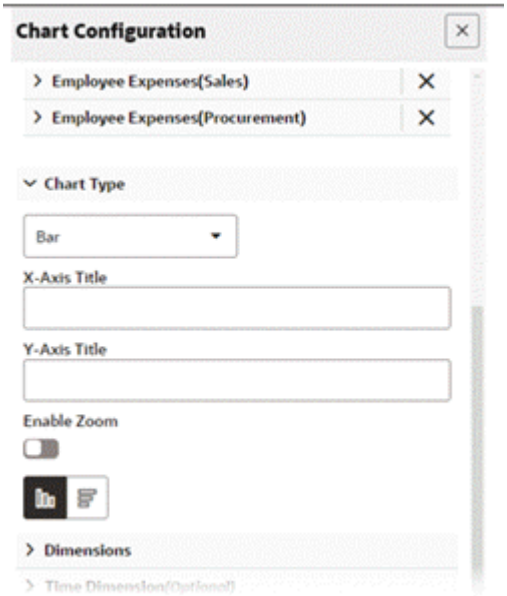
2. Define a Data Set 2 with a condition. Set a meaningful alias for the data set.

Define Second Data Set for Chart Configuration



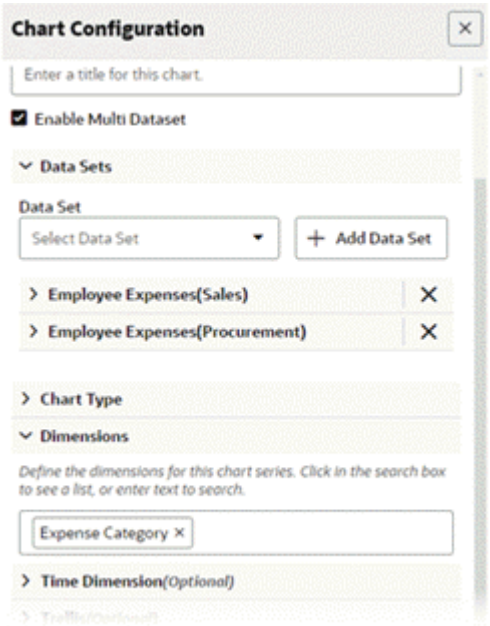
3. Set the chart type and its associated settings. Only Bar and Bar/Line charts support the multi data set feature.

Configuration of the Chart Type



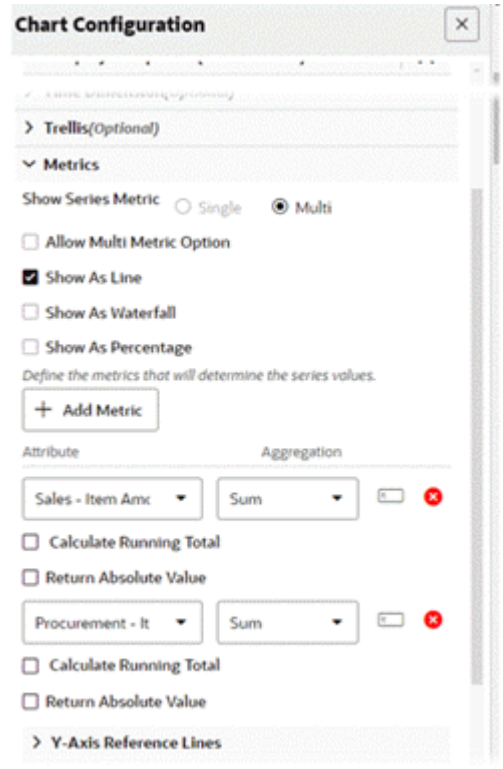
4. Set the common dimension across data sets.

Configuration of a Common Dimension across Data Sets



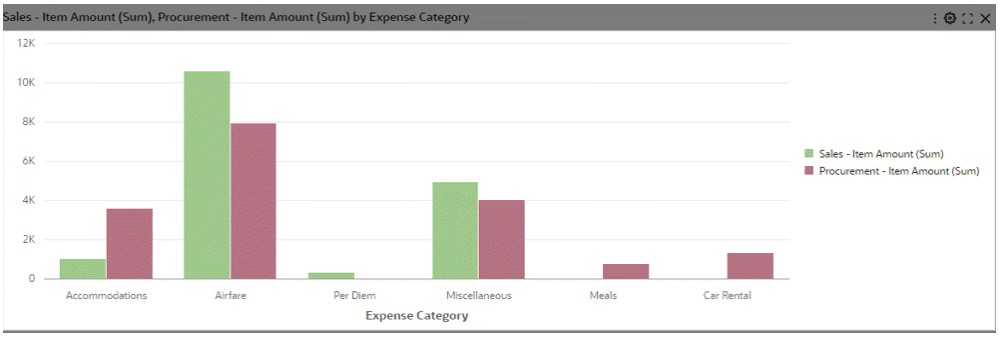
5. Set the metrics for both data sets.

Configuration of Metrics for Both Data Sets



6. Preview the chart.

Preview of a Bar Chart with Multi Data Set Support



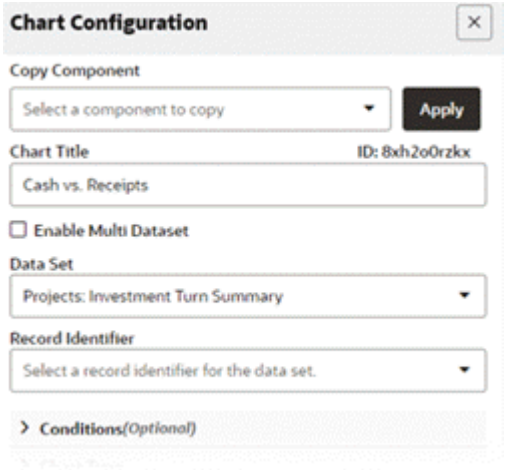
Waterfall Chart:

Waterfall Chart Configuration Options

Option	Description
Show as Waterfall	Converts a bar chart to a waterfall chart. Note that the Group dimension should not be configured because the waterfall chart does not honor stacking.
Show beginning and ending totals	Displays totals in the chart. This option also enables a runtime option for the user to control the display of the total bars.
Color Pinning	This option controls pinning colors for the increase bar, decrease bar, and total bars. Color pinning in configuration is enabled by default.

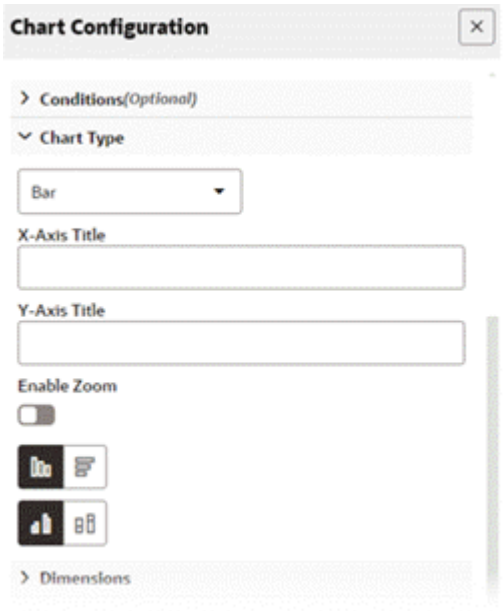
1. Define the chart title and data set.

Configuration of Chart Title and Data Set



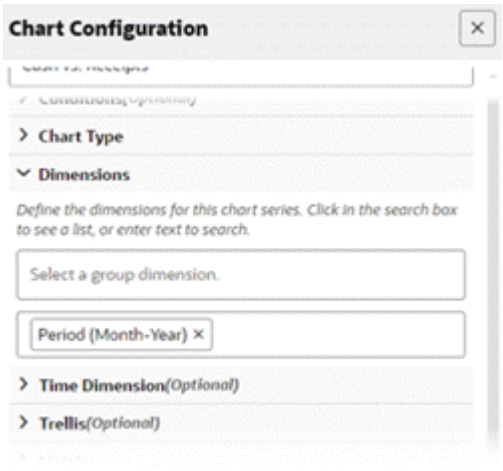
2. Define the chart type.

Configuration of Chart Type



3. Set the series dimension.

Setting the Series Dimension

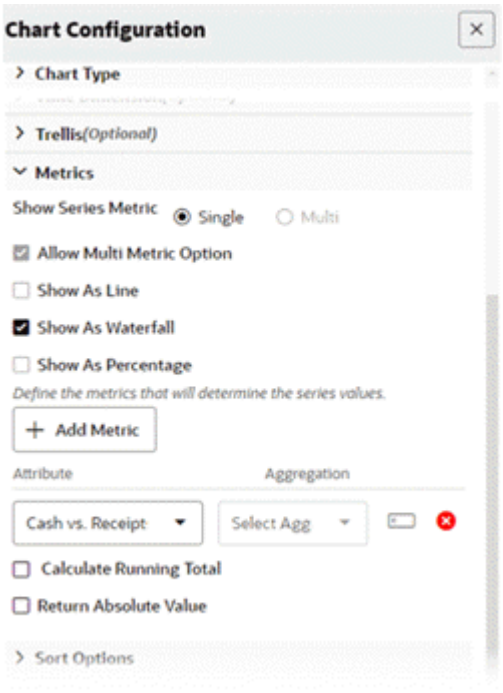


4. Set the metric.

   Check **Show as Waterfall**.

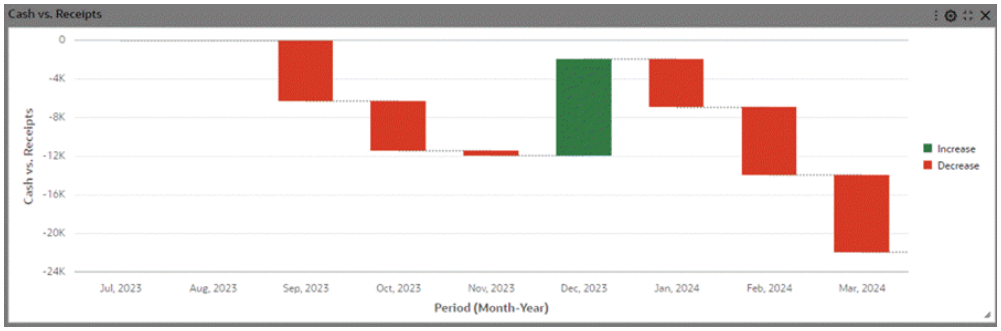
   In the following figure, the "Cash Vs Receipt" attribute is a calculated attribute. Therefore, there is no aggregation defined on it.

Setting the Metrics Configuration



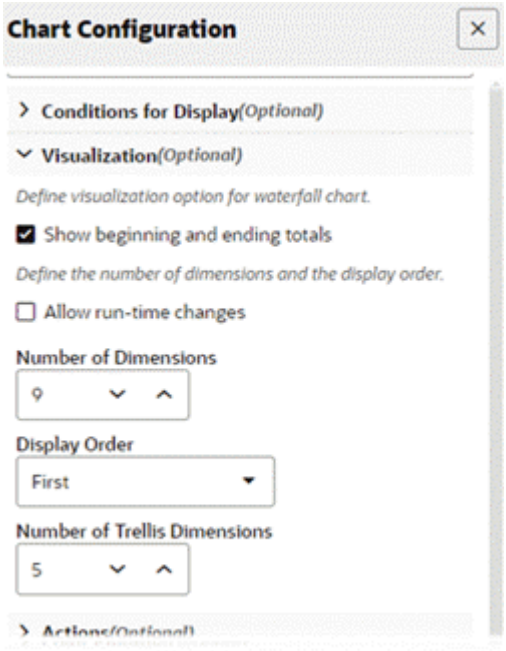
5. Preview the chart.

Preview of a Waterfall Chart



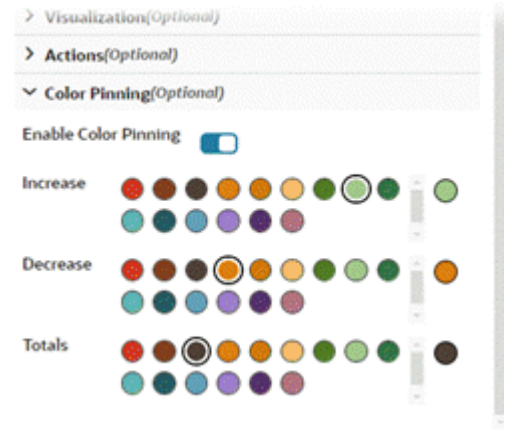
6. Inside the Visualization accordion in the Chart Configuration window, select the **Show beginning and ending totals** checkbox.

Selection of the "Show Beginning and Ending Totals" Box



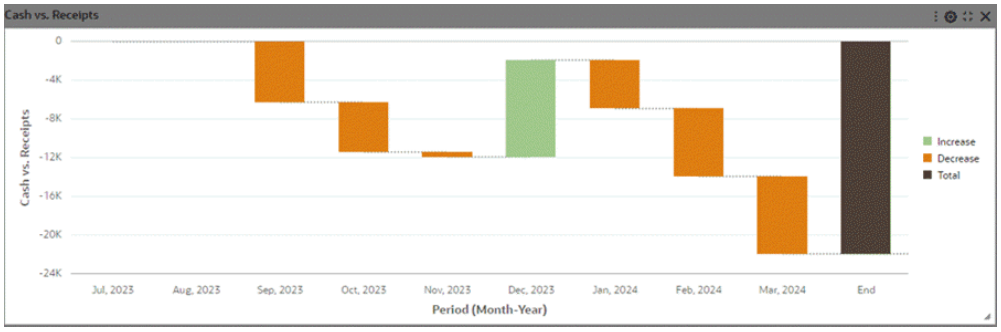
7. Enable color pinning.

Enabling Color Pinning



8. Preview the chart again.

Preview of A Waterfall Chart with Totals and Color Pinning



Percent Chart:

1. Define the chart title and data set.

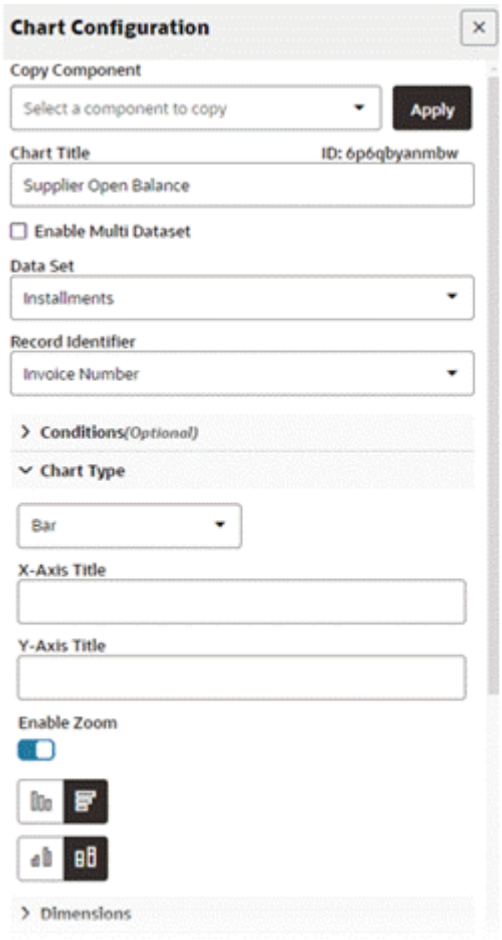
   Define the record identifier.

   Set the chart type as Bar.

   Set the orientation.

   > **Note:** Most of the data sets are stored in a denormalized form. This means that if there are two holds for a single invoice, there would be two records for that invoice. Every other attribute may remain the same. In such cases, if we don't define a record identifier and try to calculate Invoice amount per Supplier, we may end up in miscalculating the invoice amount because an invoice may appear more than once in denormalized form. Hence, if we define a record identifier, which is Invoice Number in this example, only one record corresponding to that invoice will be selected at random.

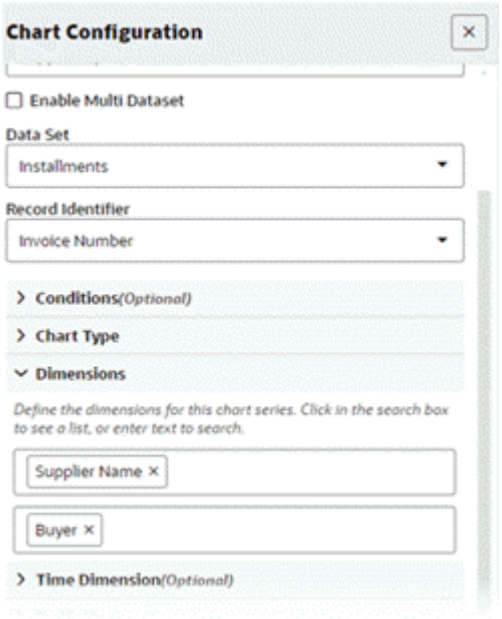
Configuration of Chart Title, Data Set, Record Identifier, Type, and Orientation



2. Define the group and series dimensions.

   Note: A percent chart only works in a stacked setup. Therefore it is necessary to define both the group and series dimensions.

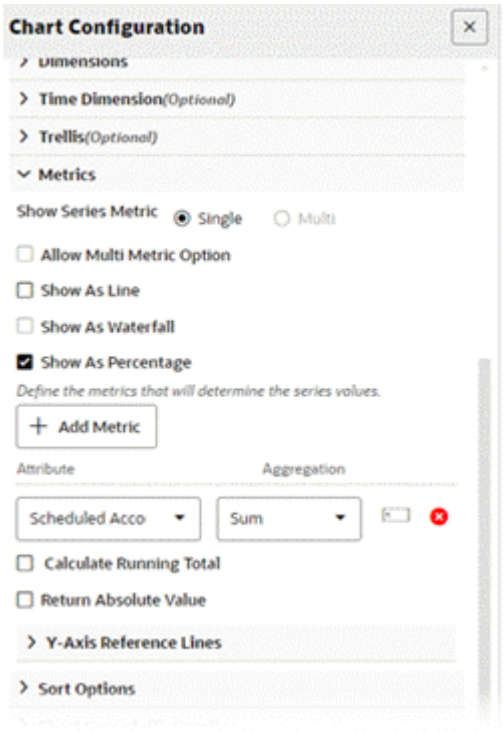
Configuration of Dimensions



3. Set metrics.

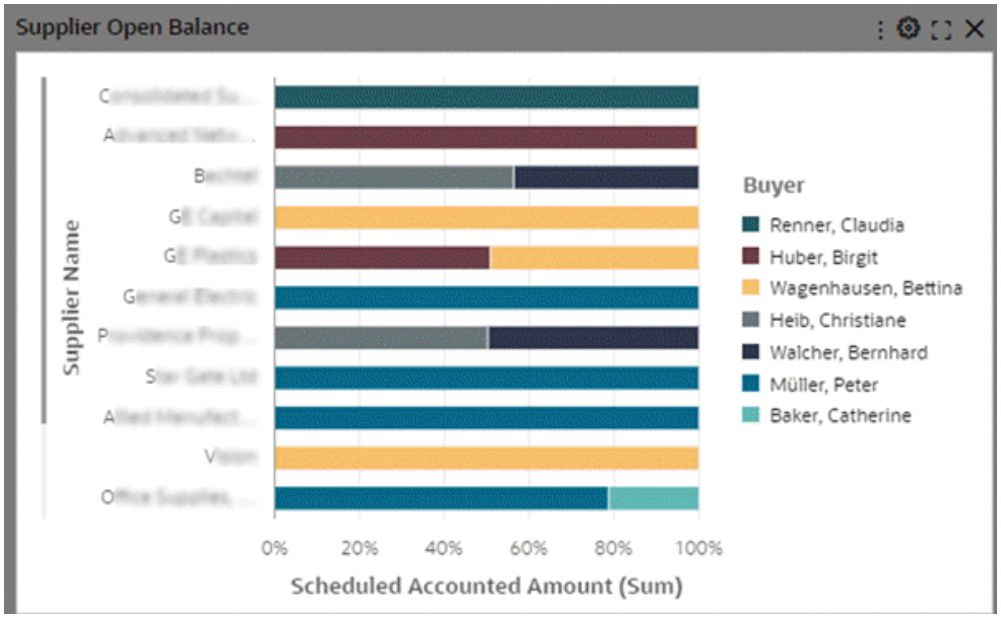
   Select the **Show as Percentage** checkbox.

Configuration of Metrics



4. Preview the chart.

Preview of a Percent Chart



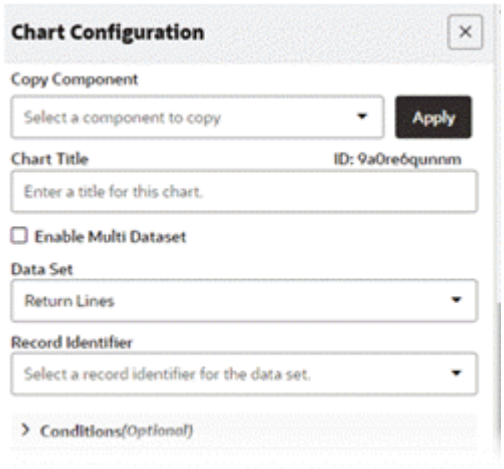
Time Series Chart:

Time Series Chart Configuration Options

Option	Description
Time Dimension	To define any date or date-time attribute as a time dimension. The date or date-time attribute must have been configured as a dimension.
Time Grain	To set the aggregation for the time dimension.

1. Define the data set.

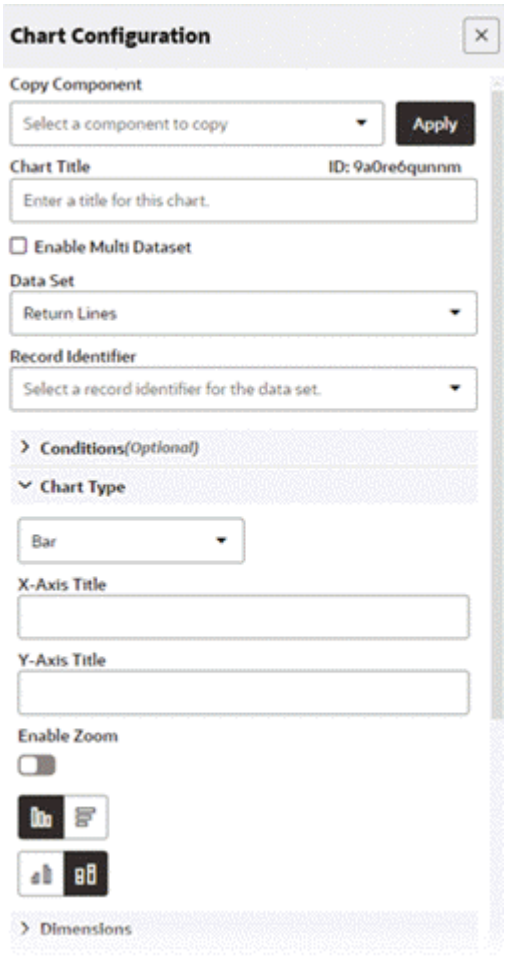
Configuration for Data Set



2. Define the chart type.

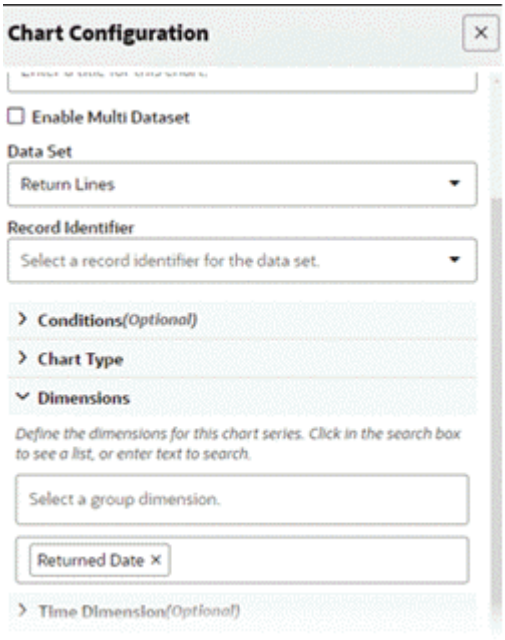
   Note that time series charts are supported in bar and bar/line charts only.

Configuration of Chart Type



3. Set a date or date-time attribute as a dimension.

Configuration of Dimension



4. Explicitly declare the dimension as a time dimension.

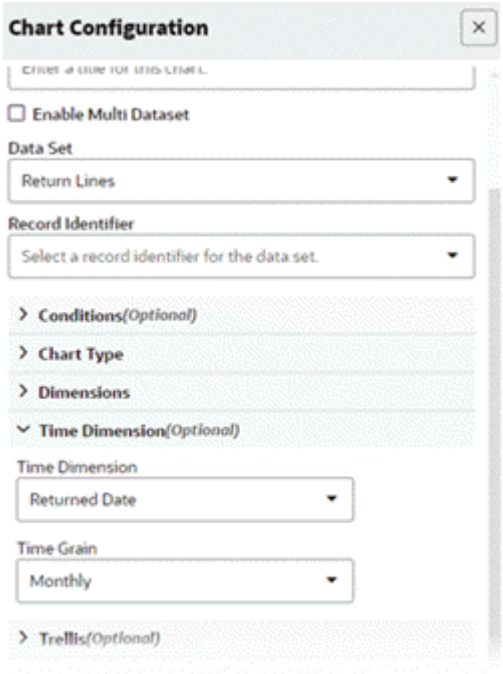
   Set the time grain.

   Note: Available time grains are:

   - Daily
   - Weekly
   - Monthly
   - Quarterly
   - Yearly

   For a date-time attribute, the hours and minute label time grain is supported only during cascading. Cascading examples are mentioned later in this section.

Declaration of Time Dimension and Setting of the Time Grain



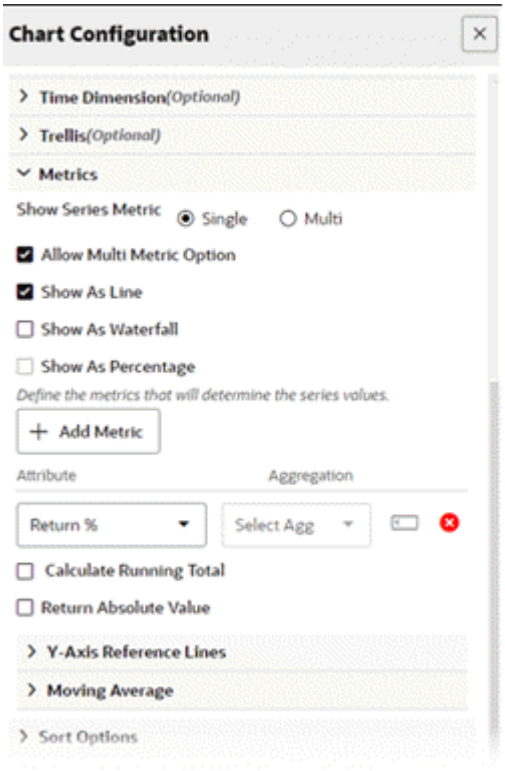
5. Set the metric.

   Select the **Show as Line** checkbox.

   - Return % is a calculated attribute and hence aggregation is not defined on it during configuration. Aggregation is already defined in metadata.

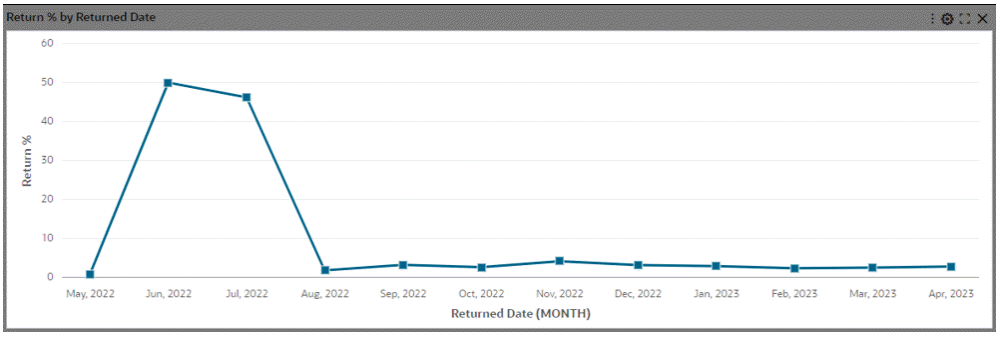
   - **Show as Line** is an optional setting here. It is more meaningful to see time trends in a line chart rather in a bar form.

Metrics Configuration



6. Preview the chart.

Preview of a Time Series Chart



Additional Configuration for Smoothing a Line Chart

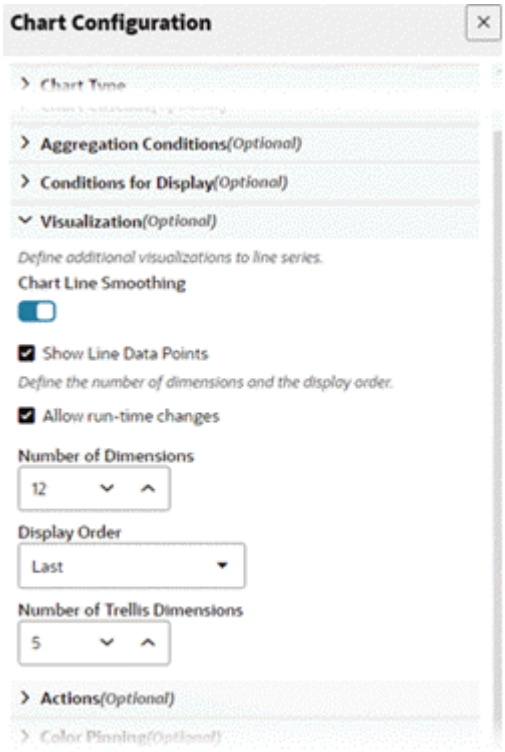
Chart Visualization Options

Option	Description
Chart Line Smoothing	Enable the flag to display a smooth line with curvature. Applicable for line and bar/line charts.
Show Line Data Points	Controls the display of data points on the line. Applicable for line and bar/line charts.

7. Continue in the same example and make the relevant changes in visualization accordion to smooth the line chart.

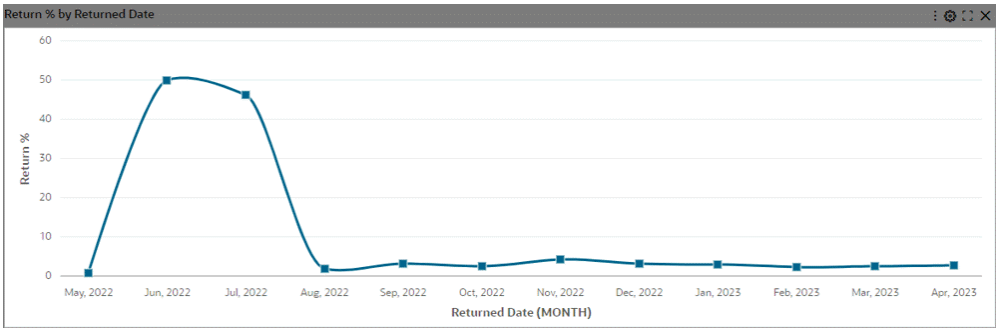
   Ensure that the **Chart Line Smoothing** checkbox is selected.

Visualization Configuration Options



8. Preview the chart.

Preview of a Chart with Line Smoothing



Examples of Configurations for Analytical Functions in Charts

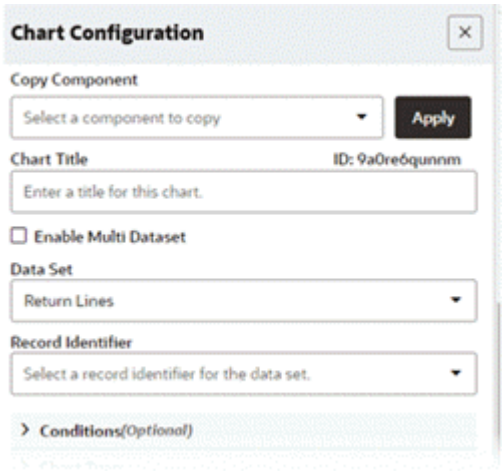
Reference Lines:

Chart Option for Reference Line

Option	Description
+ Add Y Axis Reference Line	• Reference line: allow displaying reference line on the chart based on a predefined metric. Example: Average of Asset Cost (Sum) • Can add a reference line for as many as the metrics defined on the chart. • Can add an additional reference line, Y2-Axis Reference Line, in a bar/line chart. • If the chart displays only one metric, the corresponding reference line to this metric will be displayed. • Supports average, min, max, constant. • Applied to all chart types except pie/donut charts.

1. Define the data set.

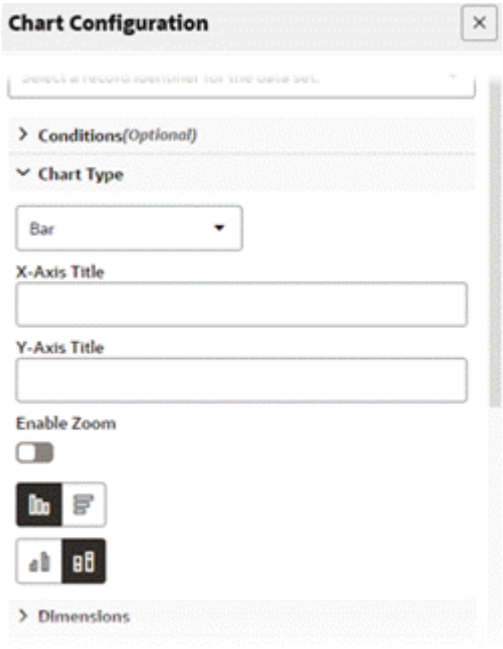
Configuration of the Data Set



2. Define the chart type.

   Time series charts are supported only in bar and bar/line charts.

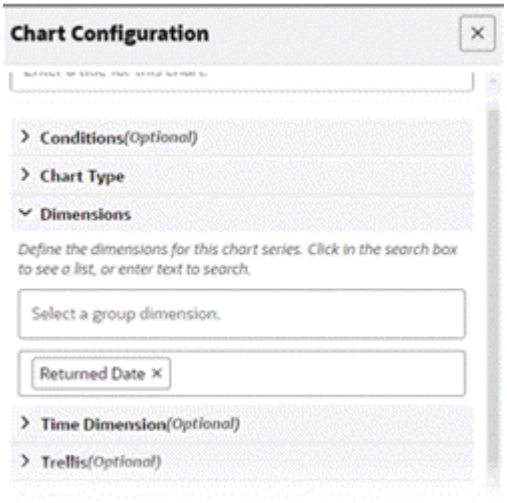
Configuration of the Chart Type



3. Set a date or date-time attribute as a dimension.

   In this example, Returned Date is set as a dimension.

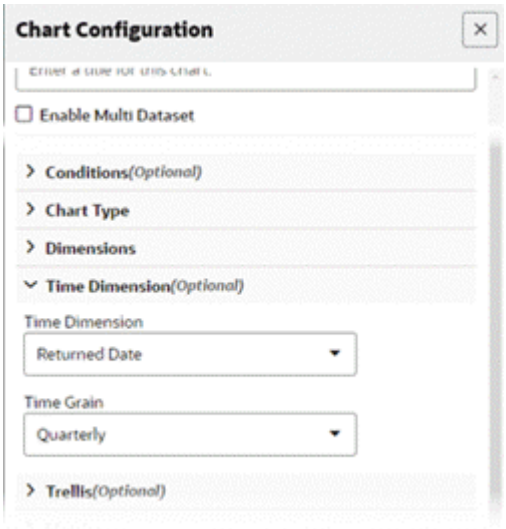
Configuration of a Date as a Dimension



4. Explicitly declare the dimension as a time dimension. In this example, Returned Date is declared as the time dimension.

   Set the time grain. In this example, the time grain is Quarterly.

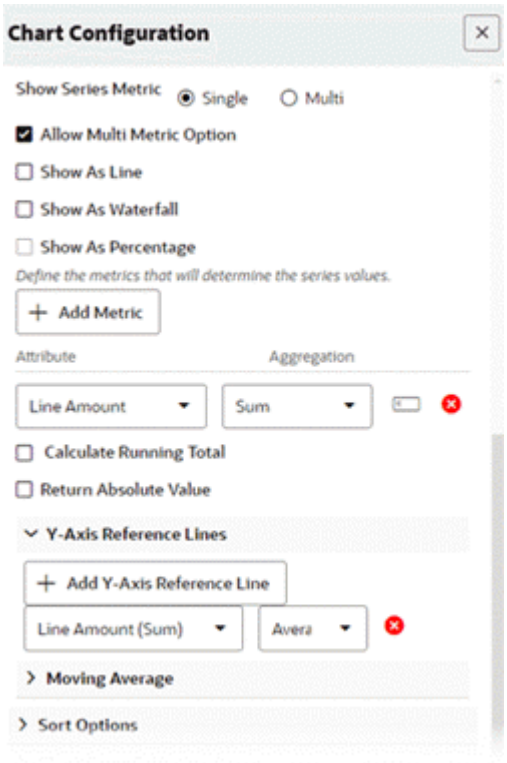
Configuration of Time Dimension and Time Grain



5. Set the metric.

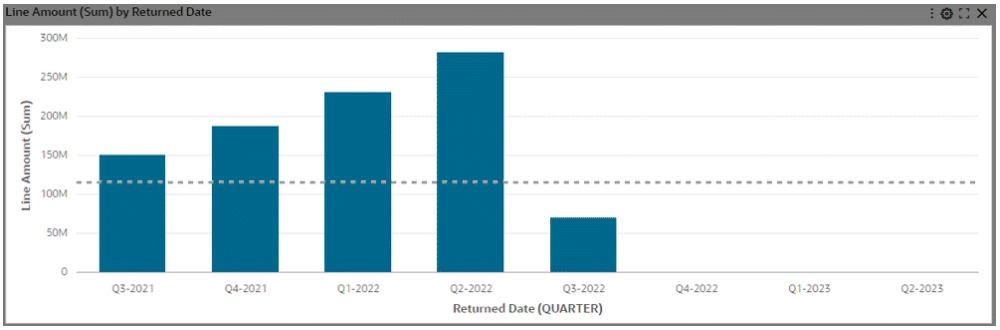
   Set the reference line to indicate an average of returned order amount per quarter.

Configuration of Metrics



6. Preview the chart.

Preview of a Reference Line in a Chart



Simple Moving Average:

Introduced in ECC V12, the simple moving average (SMA) functionality emerged as a pivotal feature for data analysis and decision-making. By smoothing trends and providing a more stable representation of data, the SMA facilitated the identification of underlying patterns and trends within time series data. Its ability to reduce the impact

of short-term volatility enabled users to make more informed decisions with greater confidence over time.

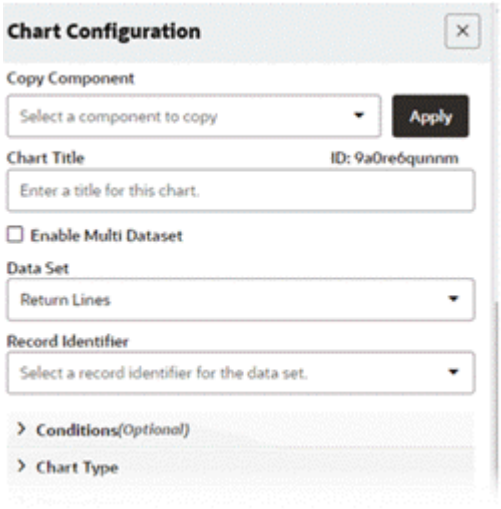
The Simple Moving Average, supported in bar and bar/line charts, is specifically designed for time series data, with the overlaid dotted line enhancing visual clarity without detracting from the original metric. Configuration of the Simple Moving Average is user-friendly, allowing for the definition of the moving average within the metric settings, adjustment of the rolling window, and the option to enable it as a default view.

Options for Simple Moving Average

Option	Description
+ Add Moving Average	Allows displaying the moving average on the chart based on a predefined metric. Users need to select the metric and define the moving average window. The default window is two (2) periods.
Show Moving Average	Enables the display of the moving average on the default view.

1. Define the data set.

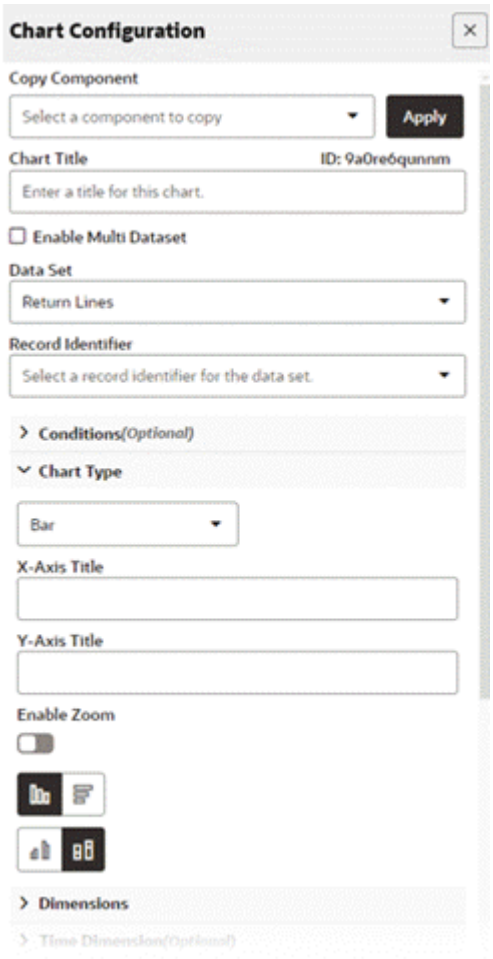
Configuration of the Data Set



2. Define the chart type.

   Note that time series charts are supported only in bar and bar/line charts.

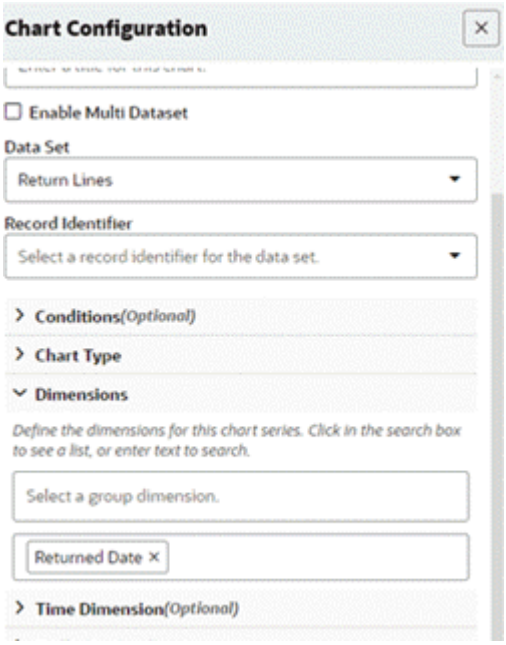
Configuration of the Chart Type



3. Set a date or date-time attribute as a dimension.

   In this example, Returned Date is set as a dimension.

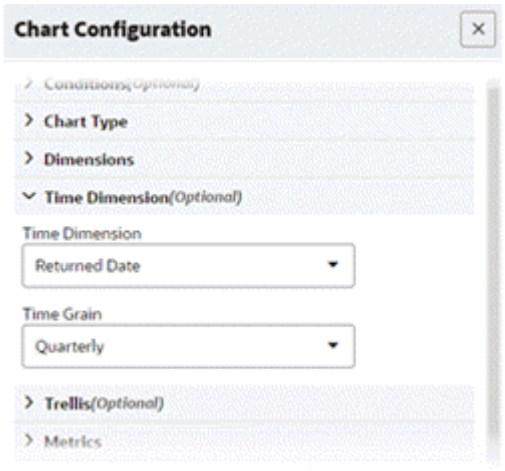
Setting the Dimension



4. Explicitly declare the dimension as a time dimension. In this example, Returned Date is declared as the time dimension.

   Set the time grain. In this example, the time grain is Quarterly.

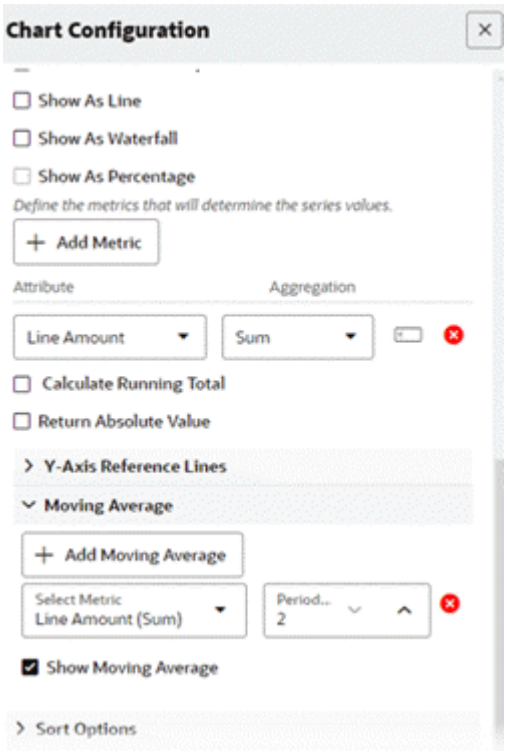
Setting the Time Dimension



5. Set the metric.

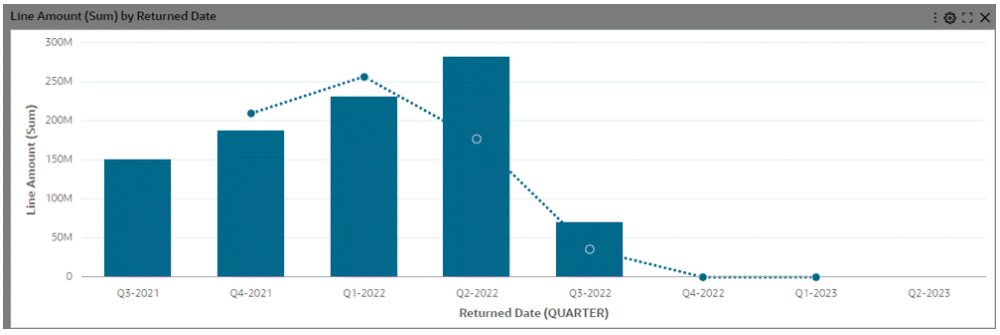
   Set the moving average period.

Configuration of Metrics Information



6. Preview the chart.

Preview of a Simple Moving Average in a Chart



Running Total:

Introduced in V12, the running total enhancement offers users a powerful tool for gaining comprehensive insights into cumulative metrics across various data aggregations. This feature, accessible through a new checkbox within the metric accordion, allows users to calculate running totals on any type of aggregation, including calculated attributes. With support extending to all bar and bar/line charts except for

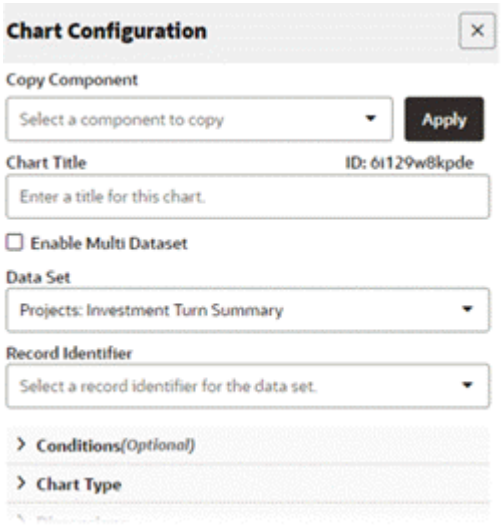
the Stacked Bar chart, users can now leverage this functionality to enhance their decision-making processes and gain deeper understanding from different perspectives.

Option for Running Total

Option	Description
+ Add Moving Average	Allows displaying running total on the chart based on a predefined metric. Designers need to define the metric and check the **Show running total** box.

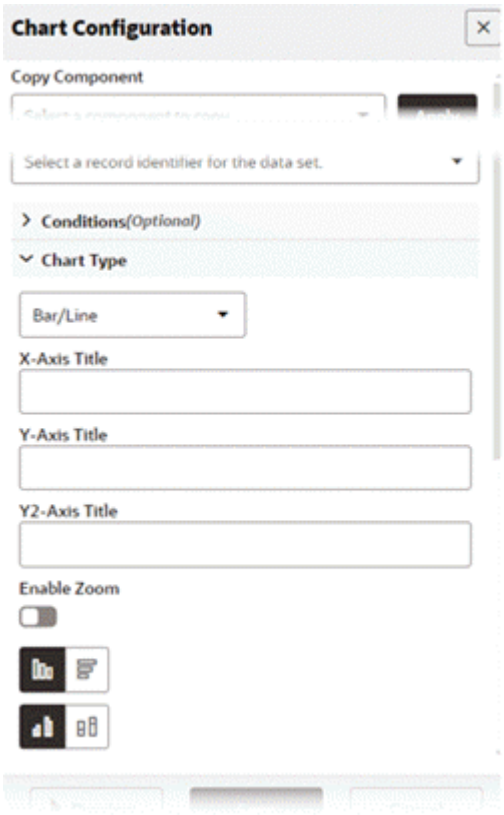
1. Define the data set.

Configuration of the Data Set



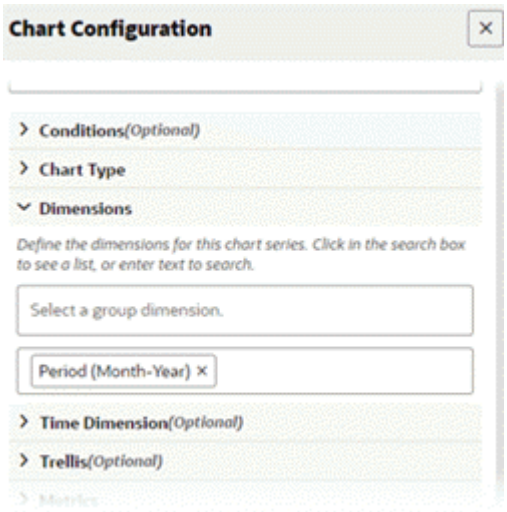
2. Define the chart type.

Configuration of the Chart Type



3. Set the dimension.

Setting the Dimension

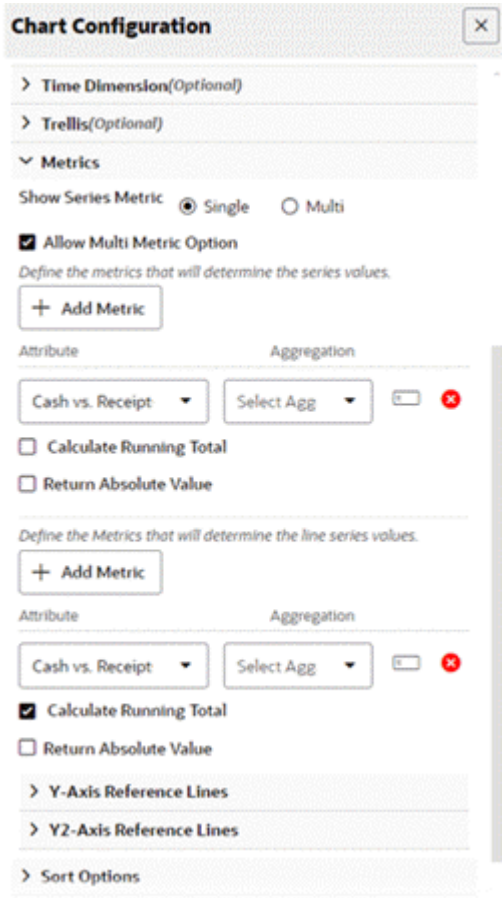


4. Set the bar metric.

   Set the line metric with a running total.

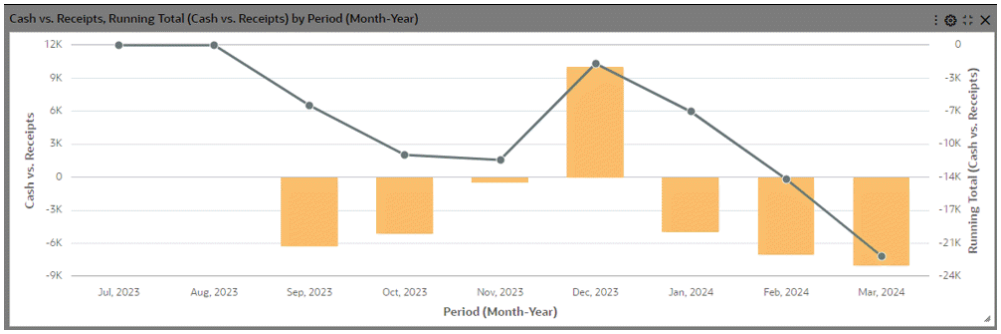
   - "Cash vs. Receipts" in this example is a calculated attribute and hence there is no aggregation.
   - In the bar metric, the metric is used without running total.
   - In the line metric, running total of the same metric is used.

Configuring the Metrics Information



5. Preview the chart.

Preview of a Running Total in a Chart

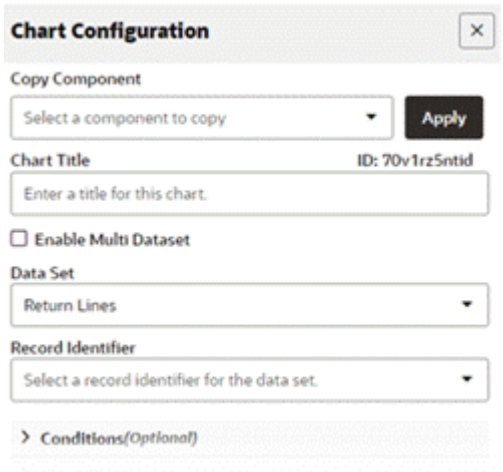


Examples of Configuration Steps for Pie and Donut Charts

Pie/Donut Chart:

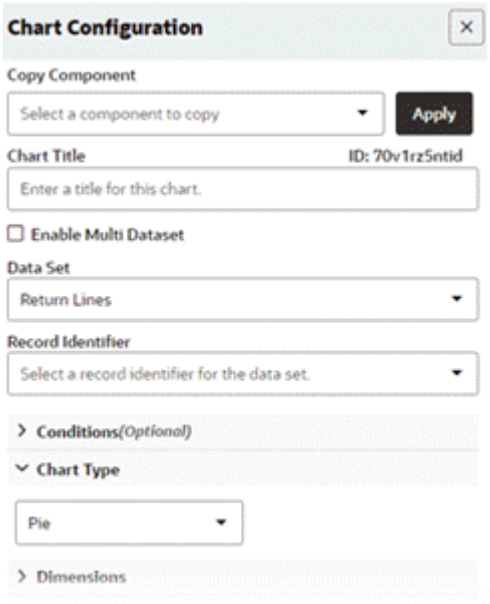
1. Define the data set.

Configuration of the Data Set



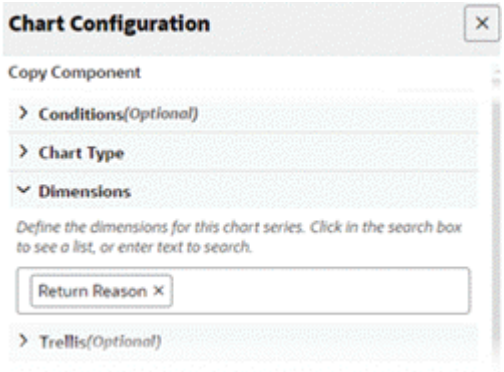
2. Set the chart type to Pie.

Configuration of the Chart Type as Pie



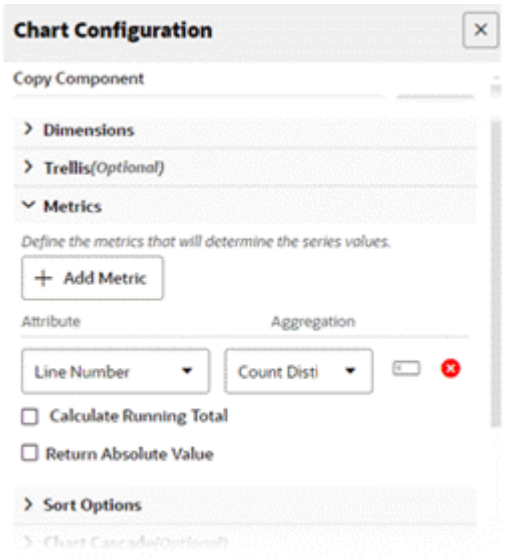
3. Set the dimension.

Configuration of the Dimension



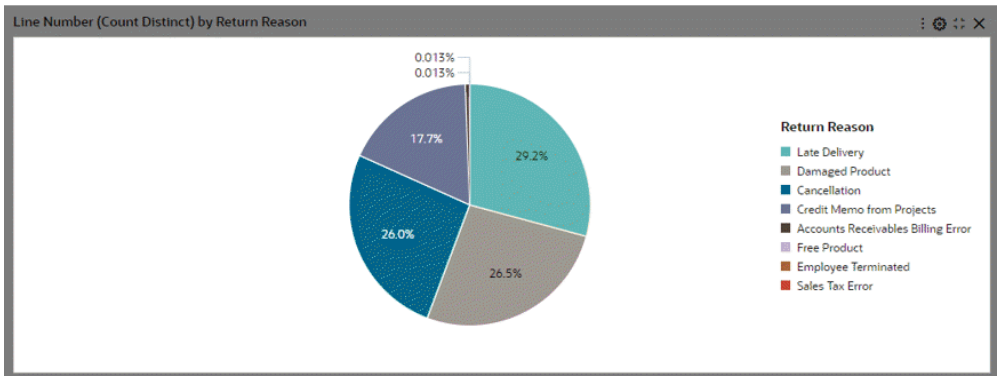
4. Set metric.

Configuration of Metric Information



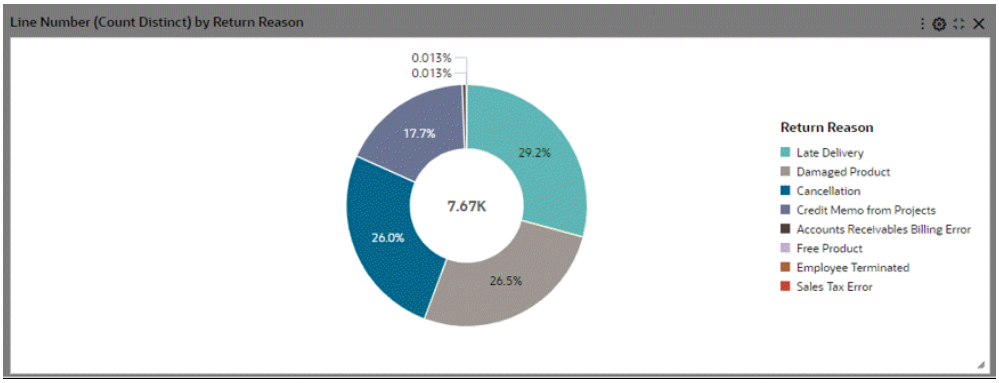
5. Preview the chart.

Preview of a Pie Chart



6. Now change the chart type in Step 2 to Donut and preview the chart.

Preview of a Donut Chart



Thresholding using the "Other" Option in Pie/Donut Charts:

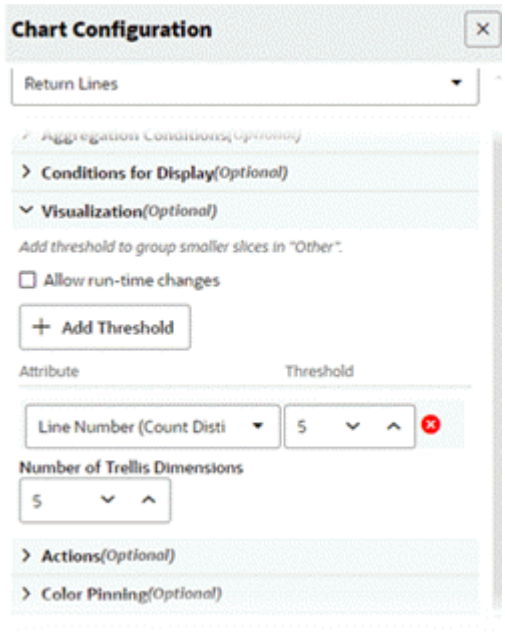
Dimensions corresponding to a percentage lower than a specified threshold are grouped into a group called "Other." If configured, the threshold limit is also controlled from runtime options. "Other" is always displayed last in the legend. A user can filter by "Other" from the chart to drill down to data grouped in the "Other" group; however, the user cannot filter by "Other" from the chart legend.

Chart Options for Thresholding

Option	Description
Allow runtime changes	Enable this flag to allow the business user to make changes to the threshold.
+ Add Threshold	Select each metric and configure threshold corresponding to that metric. The dimensions with metric values below this threshold will be grouped into "Other." Click + Add Threshold for adding a threshold to another metric.

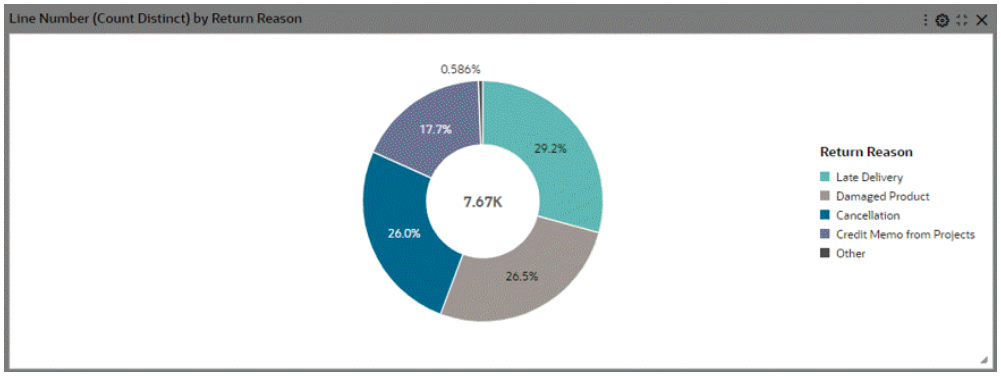
1. Configure a donut chart as described in the steps above.
2. Add a threshold. In the example below, the segments of "Return Reasons" that contribute to less than 5% of the whole will be grouped as "Other." .

Configuration of Threshold Options



3. Preview the chart.

Preview of a Chart with Thresholding



Common Chart Features

Trellis:

Introduced in V10, the support for the Trellis Chart brings forth an alternative chart type designed to enhance data visualization. The Trellis Chart presents a series of sub-charts, all utilizing the same scale and axes, thereby simplifying the comprehension of data relationships. This chart type effectively splits a single chart into multiple versions,

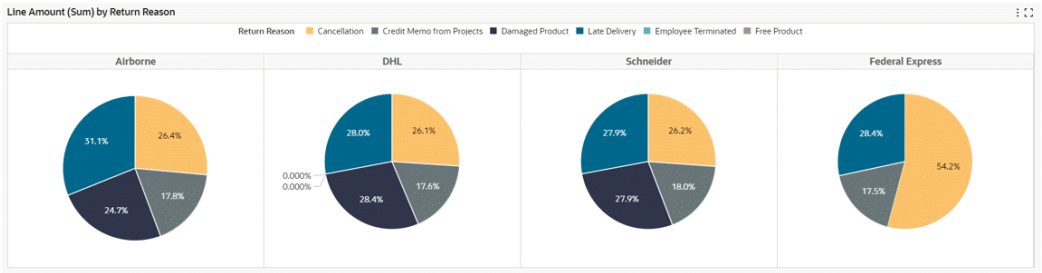
arranged side-by-side, with data partitioned across these versions based on a chosen series dimension, such as product lines or country for a "sales by category" column chart.

Chart Options for a Trellis Chart

Option	Description
Trellis Row	Allows displaying data partitioned vertically by the chosen dimension.
Trellis Column	Allows displaying data partitioned horizontally by the chosen dimension.

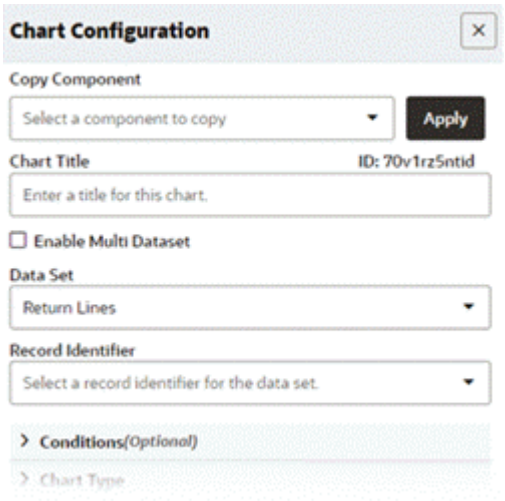
The support for pie and donut charts with trellis is introduced in V11.

Trellis Chart with Pie Charts



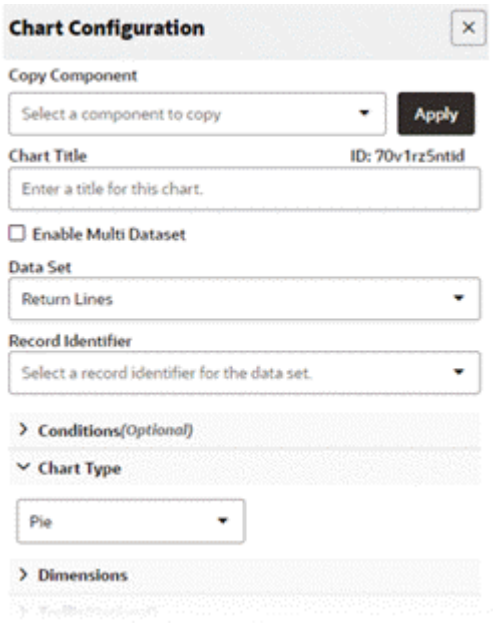
1. Define the data set.

Configuration of the Data Set



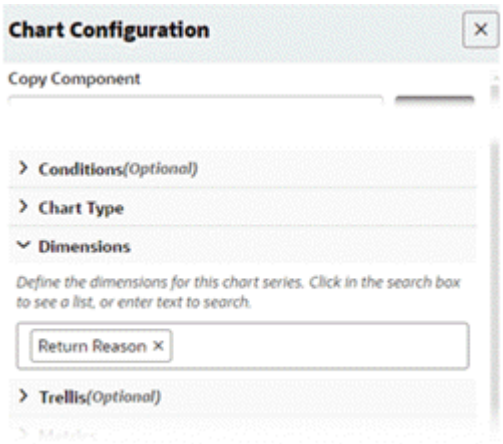
2. Set the chart type to Pie or Donut.

Configuration of the Chart Type



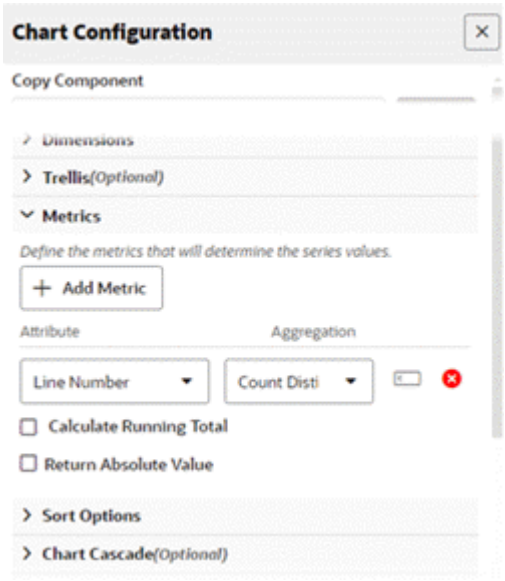
3. Set the dimension.

Configuration of Dimension



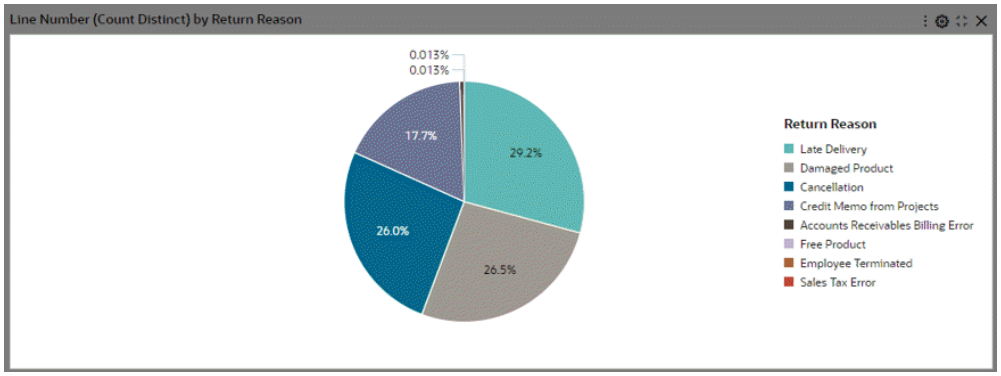
4. Set the metric.

Configuration of Metric Information



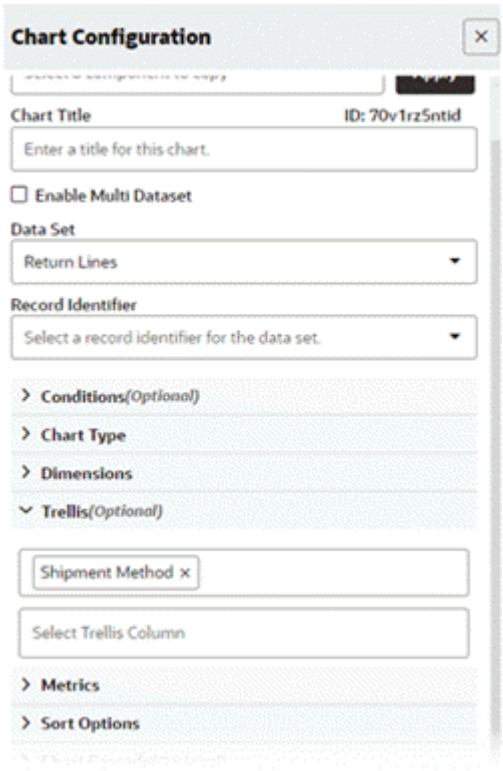
5. Preview the chart.

Preview of Pie Chart



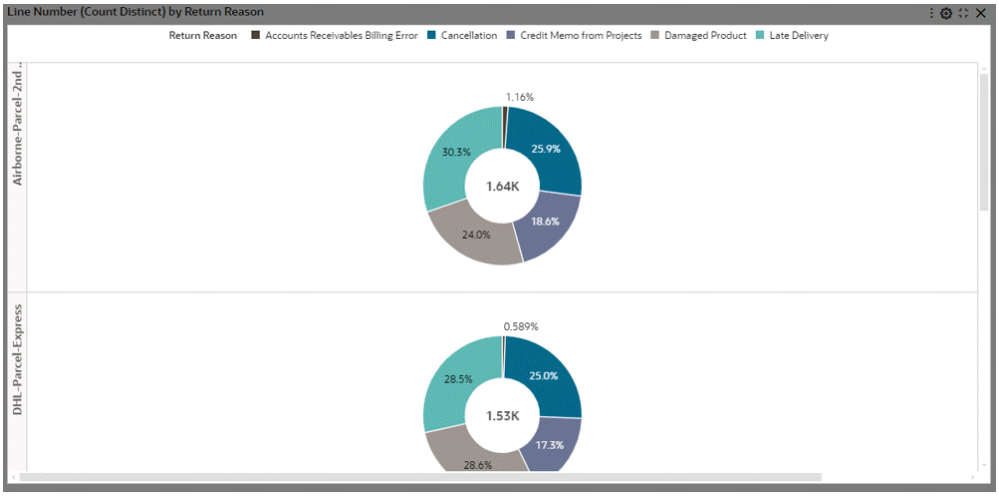
6. Add a trellis row with the attribute "Shipment Method."

Trellis Configuration with Trellis Row



7. Preview the new trellis chart.

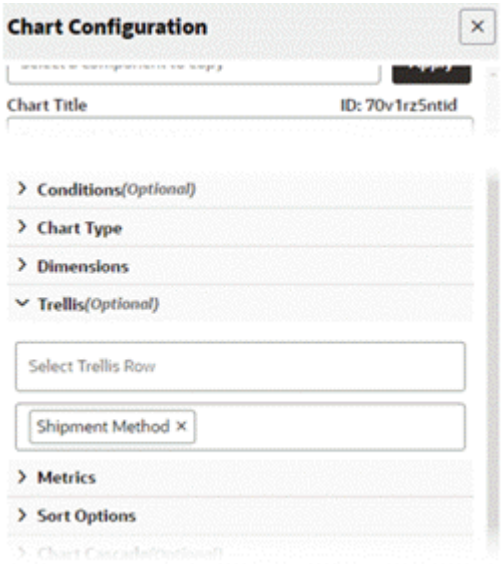
Preview of a Trellis Chart with Trellis Row Configured



8. Add a trellis column.

   Note that in this example, we have removed the attribute "Shipment Method" from Trellis Row and added it under Trellis Column.

Trellis Configuration with Trellis Column



9. Preview the chart.

Preview of a Trellis Chart with Trellis Column Configured

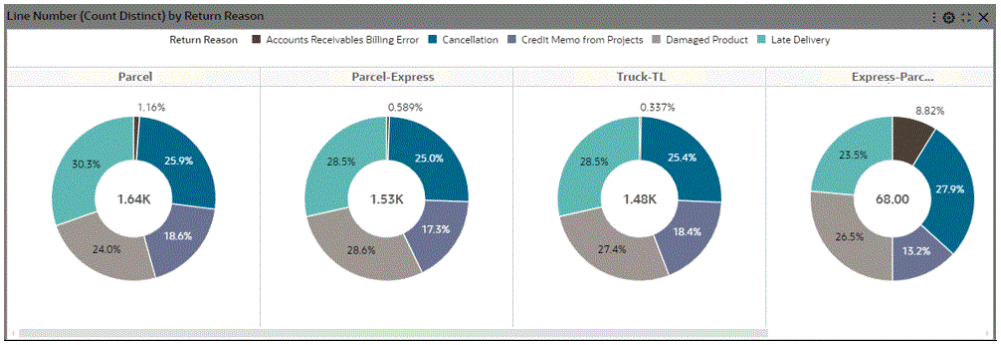


Chart Cascading Option :

Introduced in ECC V7, the enhancement of the chart tooltip empowers business users to anticipate the next level of available detail on the chart, thereby enhancing their information discovery experience. For charts with cascading enabled, users can hover over the chart series to visualize the subsequent level of details in a similar chart type.

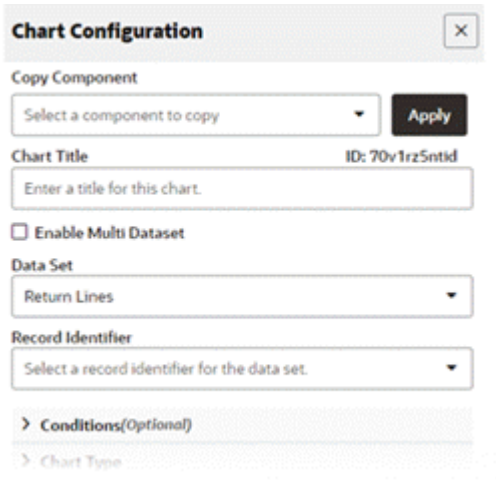
With the Cascading Breadcrumb feature, cascading levels are displayed as a trail of breadcrumbs on the top right corner of the chart, giving a better understanding of drill-downs applied to reach the current state.

Chart Cascading Options

Option	Description
Show cascading breadcrumb	Disable to not display cascading breadcrumbs.
+ Add Cascading Item	Select the first dimension (the top level of the cascade). To add another cascade level, click +Add Cascading Item and select second dimension. The new dimension is added to the end of the cascade. **Note:** To add a cascading level; all dimensions should be redefined in the **series dimension section**.

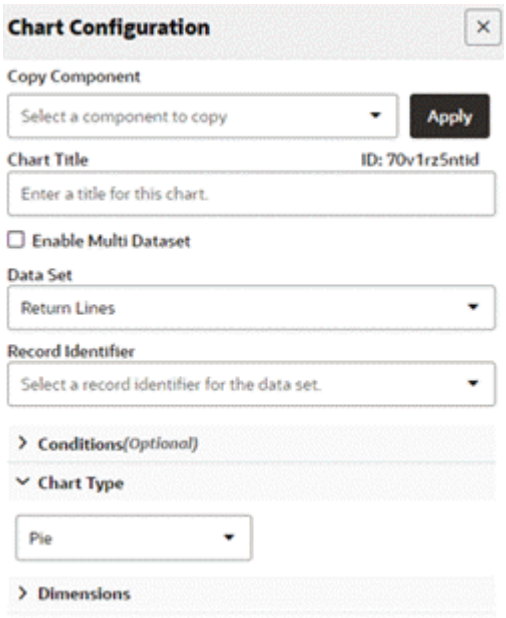
1. Define the data set.

Configuration of the Data Set



2. Set the chart type to pie or donut (donut in the example here).

Configuration of Chart Type

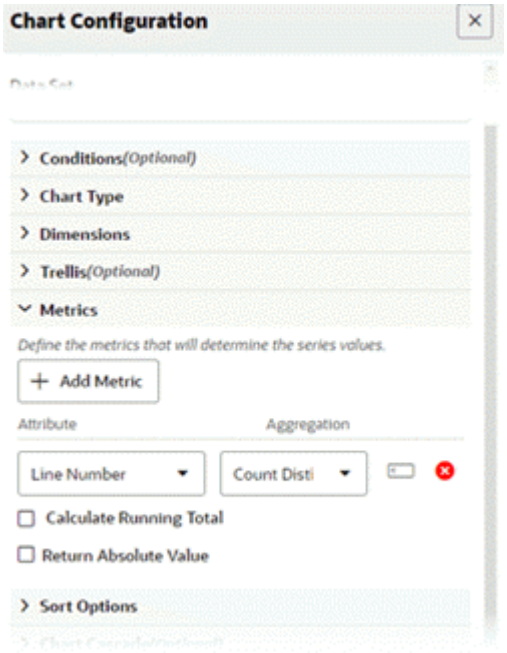


3. Set more than one series dimension to cascade.

Configuration of Dimensions

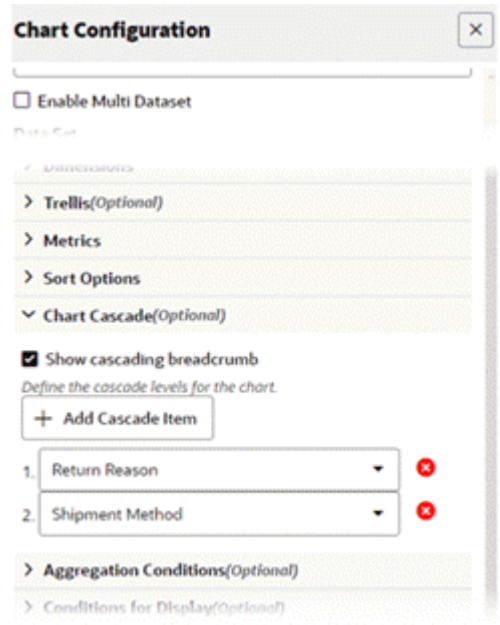
4. Set the metric.

Configuration of the Metric



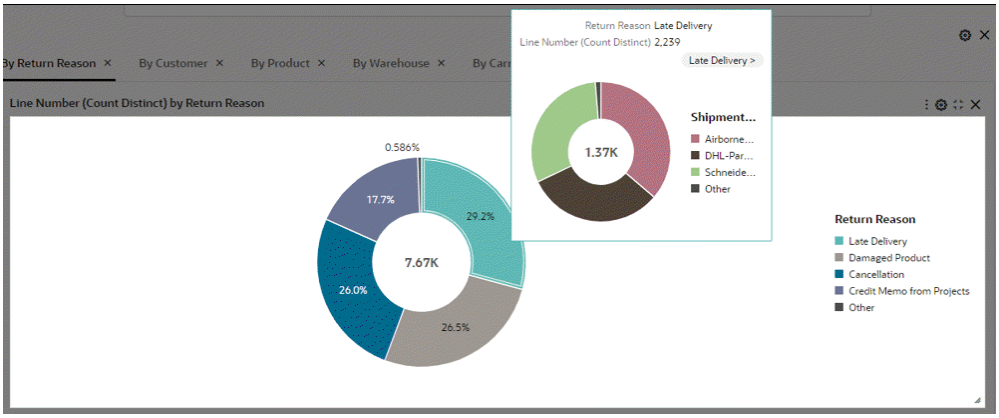
5. Define cascading.

Chart Cascade Configuration



6. Preview the chart.

Preview of a Chart with Cascading

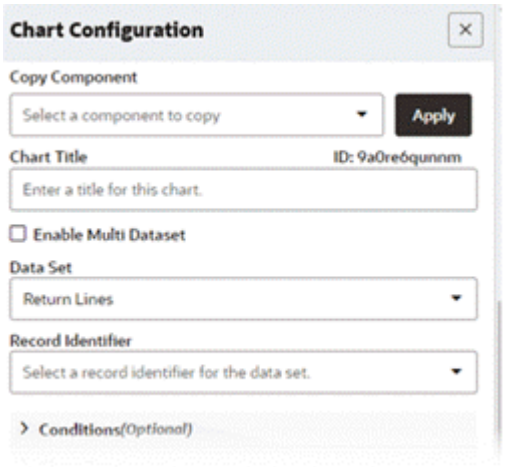


Cascading in a Time Series Chart:

In V11, the time series chart has been improved to accommodate finer time grains including Week, Day, Hour, and Minute. Hour and Minute time grains are exclusively available during cascading. The cascading drop-down list now features auto-bucketed time grains from the start point to the end, determined by the attribute profile.

1. Define the data set.

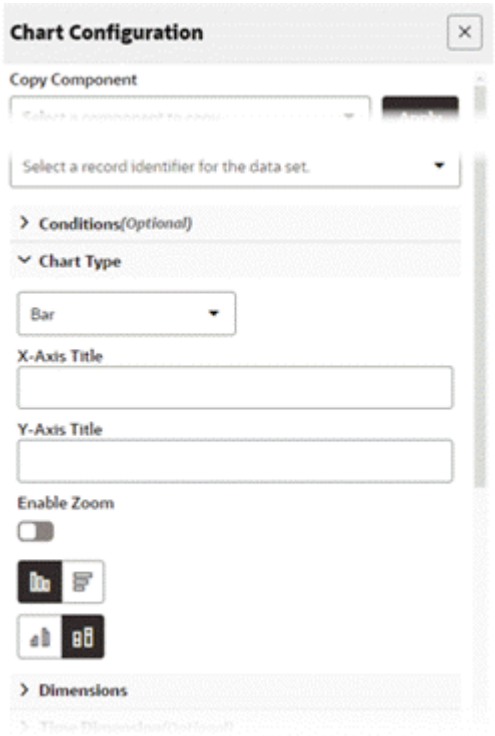
Configuration of the Data Set



2. Define the chart type. In this example, the chart type is Bar.

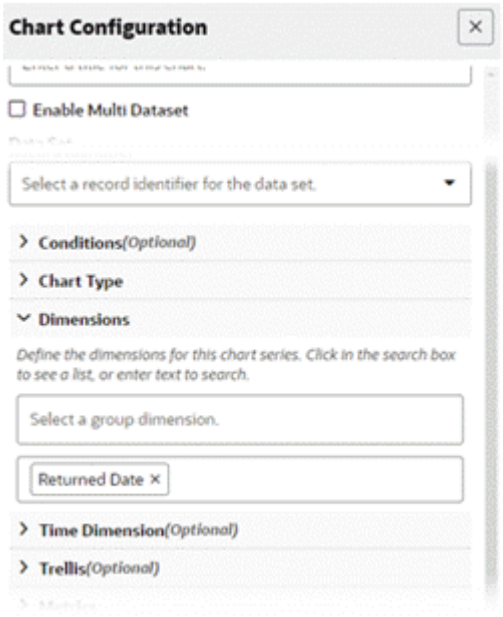
   Note: Time series charts are only supported in bar/bar-line charts.

Configuration of the Chart Type



3. Set a date or date-time attribute as the dimension.

Configuration of Dimension



4. Explicitly declare the dimension as a time dimension.

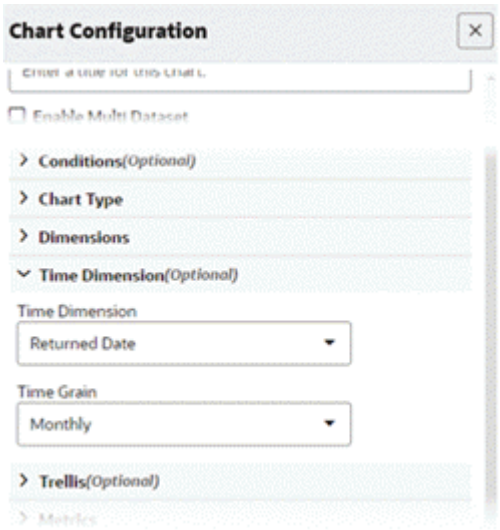
   Set the time grain.

   Note: Available time grains are:

   - Daily
   - Weekly
   - Monthly
   - Quarterly
   - Yearly

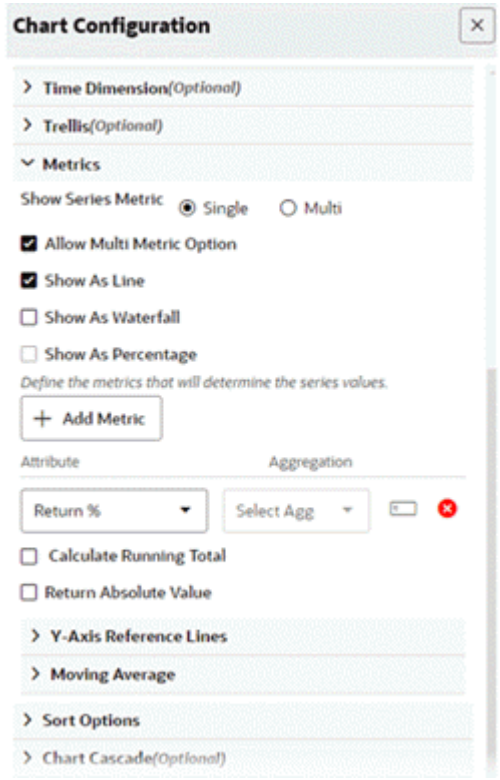
   For a date-time attribute, the hours and minute label time grain is supported only during cascading.

Configuration of Time Dimension



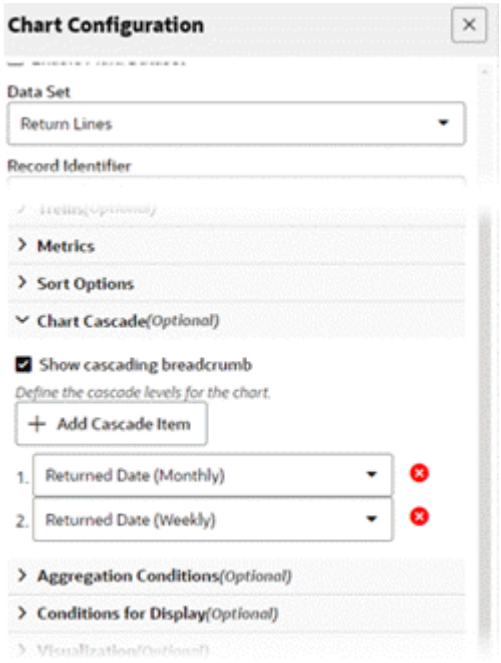
5. Set the metric.

Configuration of Metric



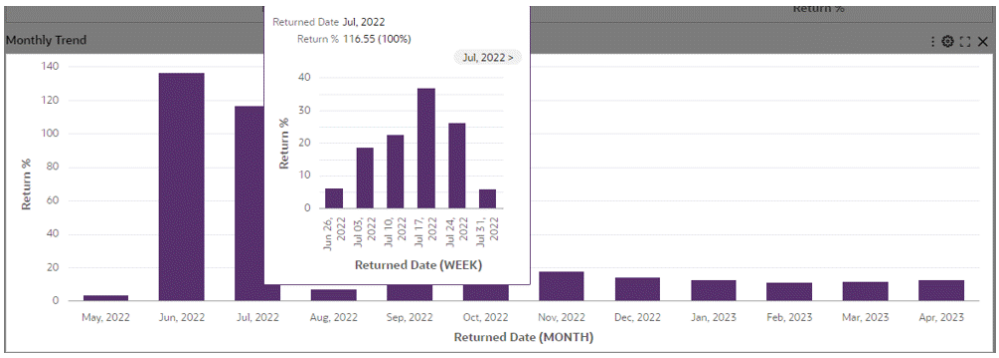
6. Set the monthly to weekly cascading.

Configuration of Chart Cascade



7. Preview the chart. Hover over any bar in previewing.

Preview of Cascading in a Time Series Chart



Sorting:

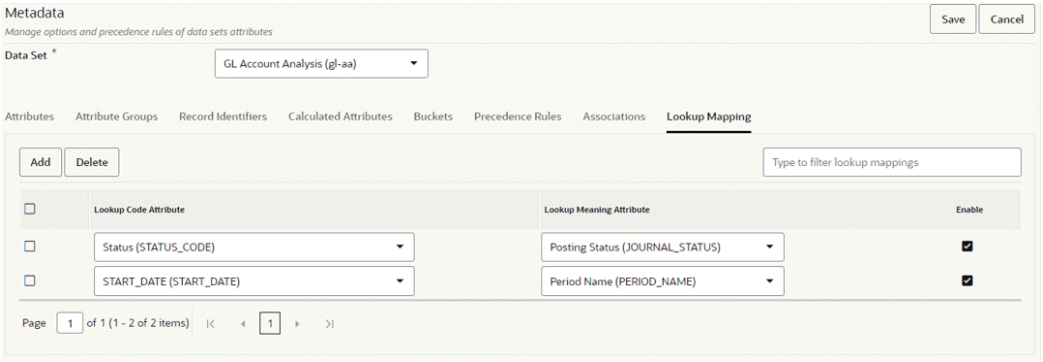
Sorting has the following options:

Options

Option	Description
Group Sort	Enable sorting either by group dimension or metric.
Series Sort	Enable sorting on series dimension.
Trellis Row Sort	To sort by the dimension configured as the trellis row.
Trellis Column Sort	To sort by the dimension configured as the trellis column.

- Sorting is not possible in a time series chart. A time series chart is always sorted by an ascending order of time dimension.
- Introduced in V11, alphanumeric sorting enhancements in ECC utilize the link between the code and the meaning attribute for sorting "alphanumeric values" across the dashboard.

Example of Sorting in the Metadata Page



Top N in Charts:

For chart types other than Pie and Donut, the Top N feature can be used to include just the essential information by rendering the chart with a subset of dimensions to display only the first N dimensions in the chart. Top N takes the Sort definition into account for considering the first or last N dimensions.

Top N is applicable for all chart types apart from pie and donut chart types, if the group

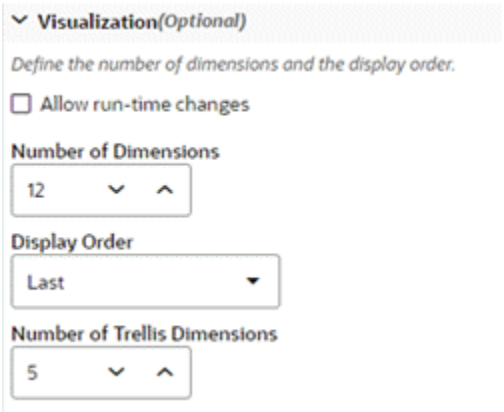
dimension is configured.

In Bar and Bar/Line chart types, if the chart has group dimension, then Top N considers the group dimension; otherwise the series dimension. In Bubble and Scatter chart types, Top N always considers the group dimension.

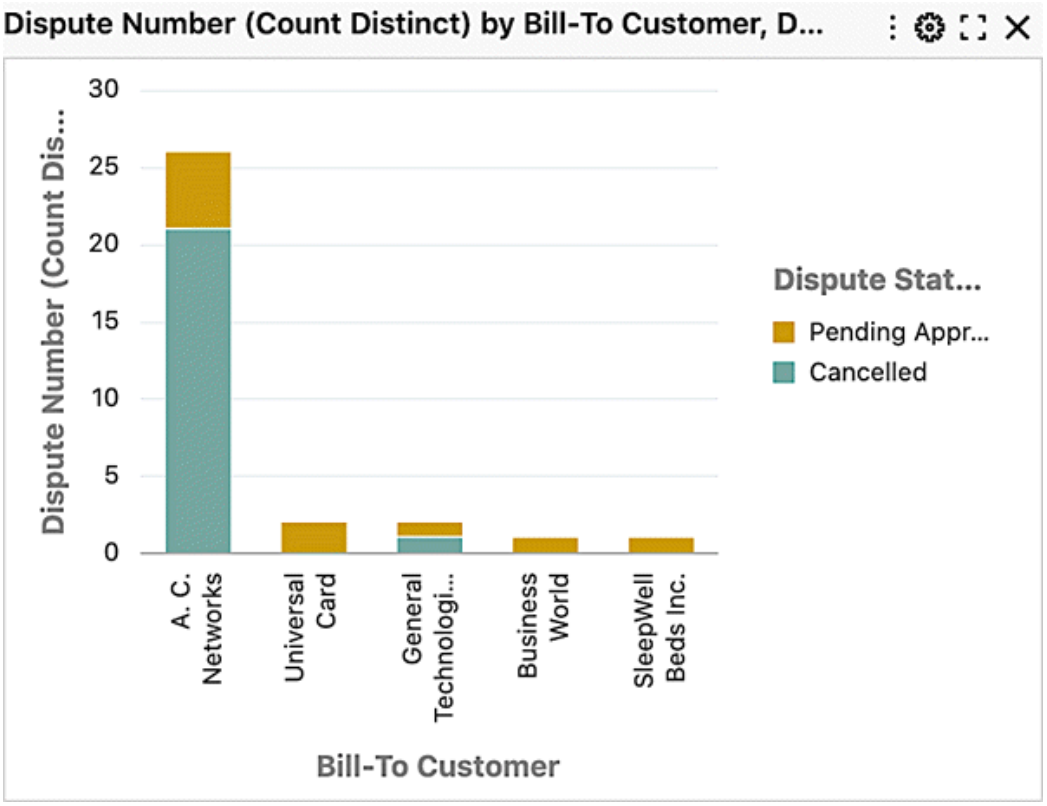
Top N Configuration Options

Option	Description
Number of Dimensions	Controls the number of dimensions displayed on the chart. If the group dimension is configured, the number of dimensions is used to control the number of group dimensions displayed; otherwise, the series dimension is used.
Order of Display	Order of display controls whether to display the first or last dimensions after considering sorting. Applicable only if sorting is configured.
Number of Trellis Dimension	Controls the number of trellis dimensions displayed on the chart. By default, it is set to five (5).

Visualization Options for the Top N Feature



Example of Top N in a Bar Chart



Export in Charts:

A chart can be exported as a PNG image that contains a snapshot of the chart. The snapshot contains the chart as-is -- as seen on the dashboard to honor runtime changes and any change in size with chart maximization.

Underlying data of the chart component can also be exported in a CSV file. Exported data also honors all the runtime changes. As the underlying data needs to be holistic, the exported data of chart will contain all the data even if Top N is configured.

Export can be configured from Actions accordion.

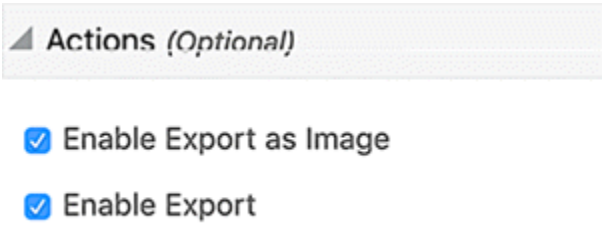
Actions Options

Option	Description
Enable Export as Image	This option allows the business user to export a snapshot of the chart. Disable this option to change the default option.

Option	Description
Enable Export	Enable this flag to allow the business user to export underlying data of the chart. Disable to change the default option.

A trellis chart cannot be exported in the form of an image.

Actions Configuration



Color Pinning in Charts :

Beginning with V6, chart colors can be pinned with context-specific colors. These colors remain intact irrespective of user session.

The following table describes the instances in which color pinning is supported: The availability for color pinning in a group dimension, series dimension, or metric plus threshold is listed for each chart type.

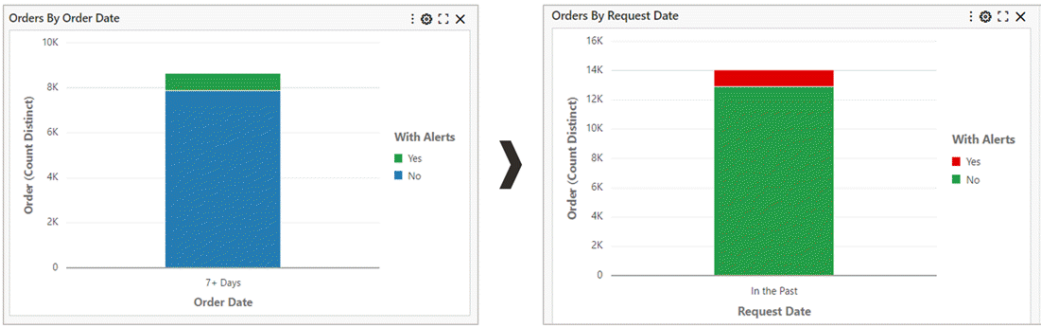
Support Matrix for Color Pinning in Charts

Chart Type	Group Dimension	Series Dimension	Metric +Threshold
Pie	N/A	Available	N/A
Donut	N/A	Available	N/A
Bar	N/A	Available	Available
Multi-Metric Bar	N/A	N/A	Available
Stacked Bar	N/A	Available	Available

Chart Type	Group Dimension	Series Dimension	Metric +Threshold
Line	N/A	Available	Available
Multi-Metric Line	N/A	N/A	Available
Stacked Line	N/A	Available	Available
Bar/Line	N/A	N/A	Available
Stacked Bar/Line	N/A	Available	Available
Scatter	Available	N/A	N/A
Bubble	Available	N/A	N/A

A color pinned on a dimension (for example, in a pie chart) honors configuration from the metadata. Colors pinned on metrics (for example, in a multi-metric bar chart) honor the component configuration.

Example of Color Pinning

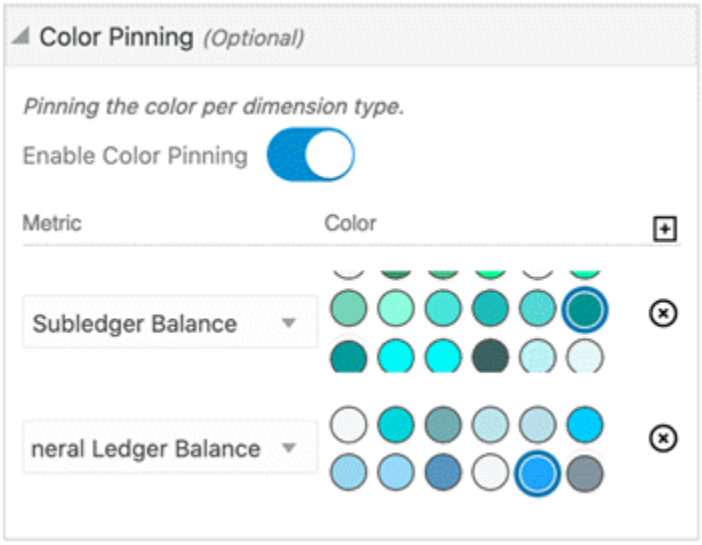


Actions Options for Color Pinning

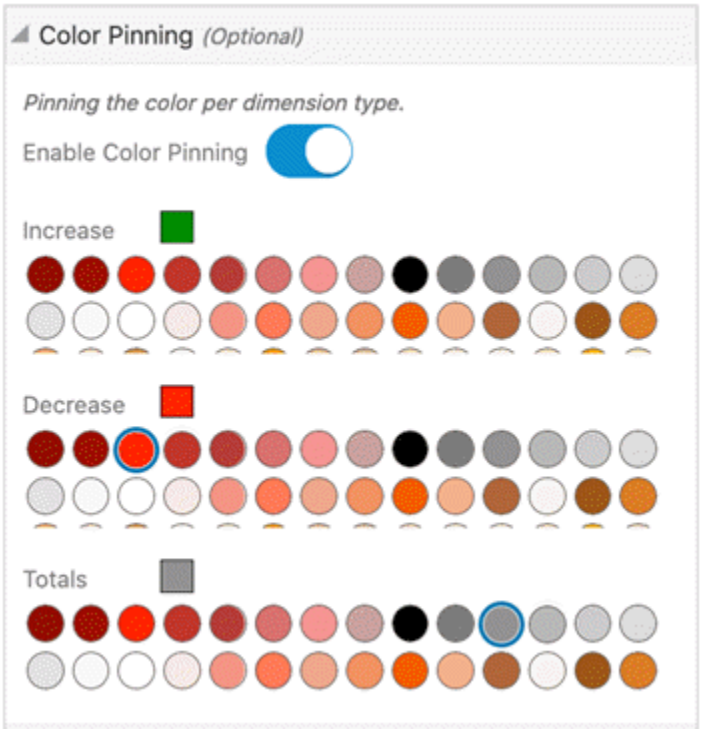
Option	Description
Enable Color Pinning	This control enables configuration for color pinning.
Add Color	This option adds an option for color pinning on a metric.

Option	Description
Select Metric	Select one of the configured metrics.
Select Color	Select one of the colors in color palette.
Increase, Decrease, Totals	This option applies to the waterfall chart only. Select colors for each change in waterfall chart.
Preview Color	This option applies to the waterfall chart only. The selected color appears for preview.

Example of Actions Configuration for Color Pinning



Example of Actions Configuration for a Waterfall Chart



Introduced in ECC V10, a new color palette has been introduced for dataset color, summary bar, flag color, and metric colors in components:

Color Palette for Selected Components

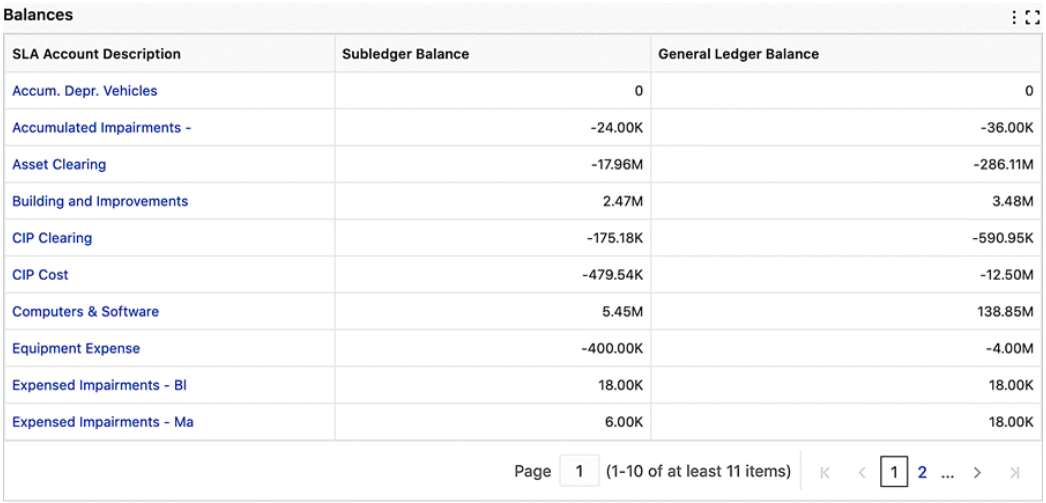
In addition, a new color palette is introduced for color pinning in metadata, which will then reflect changes in components such as the Results Grid, and the Results Table.. The following is the color palette for color pinning in metadata:

Color Palette for Color Pinning in Metadata

Aggregated Table

An aggregated table tabulates aggregations across dimensions of interest. The displayed metrics are aggregations defined at design time. The aggregate table displays ten records per page, but when maximized, it displays up to 50 records per page. The aggregated table also displays abbreviated values with actual values displayed in the tooltip. The abbreviation is also language-sensitive.

Example of an Aggregated Table



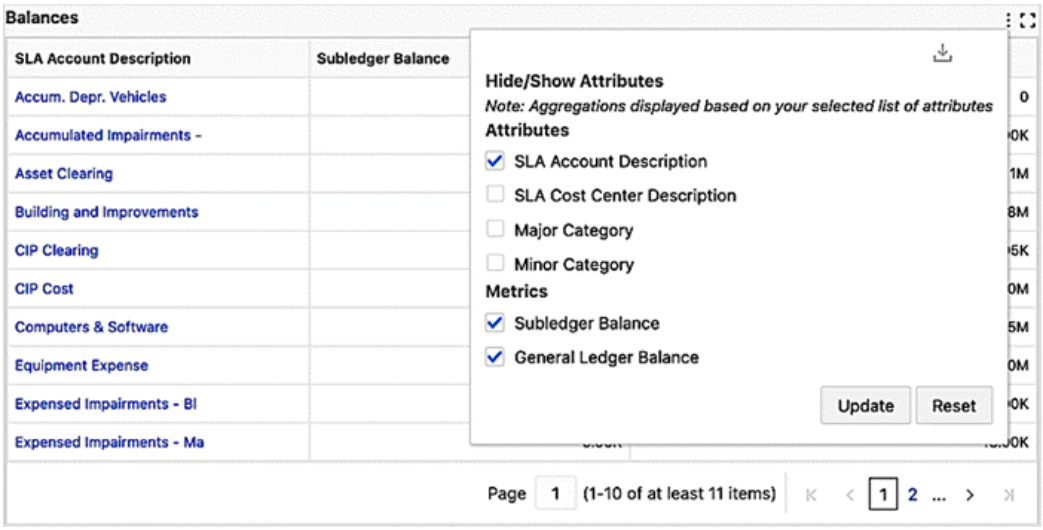
Balances

SLA Account Description	Subledger Balance	General Ledger Balance
Accum. Depr. Vehicles	0	0
Accumulated Impairments -	-24.00K	-36.00K
Asset Clearing	-17.96M	-286.11M
Building and Improvements	2.47M	3.48M
CIP Clearing	-175.18K	-590.95K
CIP Cost	-479.54K	-12.50M
Computers & Software	5.45M	138.85M
Equipment Expense	-400.00K	-4.00M
Expensed Impairments - BI	18.00K	18.00K
Expensed Impairments - Ma	6.00K	18.00K

Page 1 (1-10 of at least 11 items) 1 2 ...

A business user can adjust the data displayed in the aggregated table from a runtime option.

Example of Adjusting Data for the Aggregated Table Runtime Option



Specific Configuration

Specific Configuration Options for an Aggregated Table

Option	Description
Add Attribute	Click + Add Attribute The attribute list contains the dimensions used to aggregate the metrics. Dimension columns are always displayed to the left of the table. Beginning with V10, conditional formatting for attributes is driven by metadata colors.
Add Metric	Click + Add Metric The metric attribute has a default aggregation function assigned per attribute type. You can change the default aggregation. Beginning with V10, the metric attribute also has a provision to set conditional formatting based on the metric value.

Option	Description
Show as abbreviated number	This flag controls the display of aggregated metrics as abbreviated values instead of actual values. Actual values are displayed as a tooltip. You can disable the flag to change the default option.
Actions	Enable Export This flag controls the ability of the business user to export the data in the aggregate table. Export honors granularity in the aggregate table by including all the selected attributes and metrics. You can disable the flag to change the default option.
Enable Aggregated Table Summary	Enable displaying summary of all metrics. The summary is impacted by selected refinements and local filters.

Export for Aggregated Tables

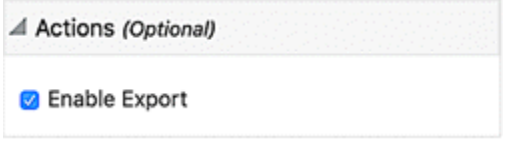
Enable the Export option to export the contents of the aggregated table. Export considers granularity of the aggregated table set from runtime options.

Export can be configured from the Actions accordion.

Actions Option

Option	Description
Enable Export	Controls whether a business user can export the underlying data in the aggregated table. Disable to change the default option.

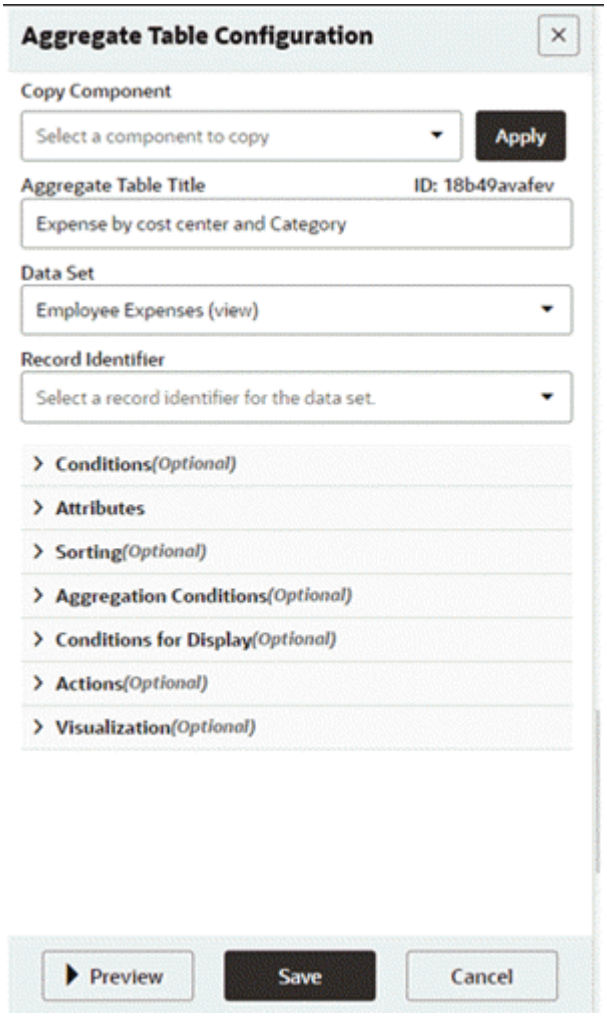
Aggregated Table Actions Option



The following steps outline the configuration of an aggregated table.

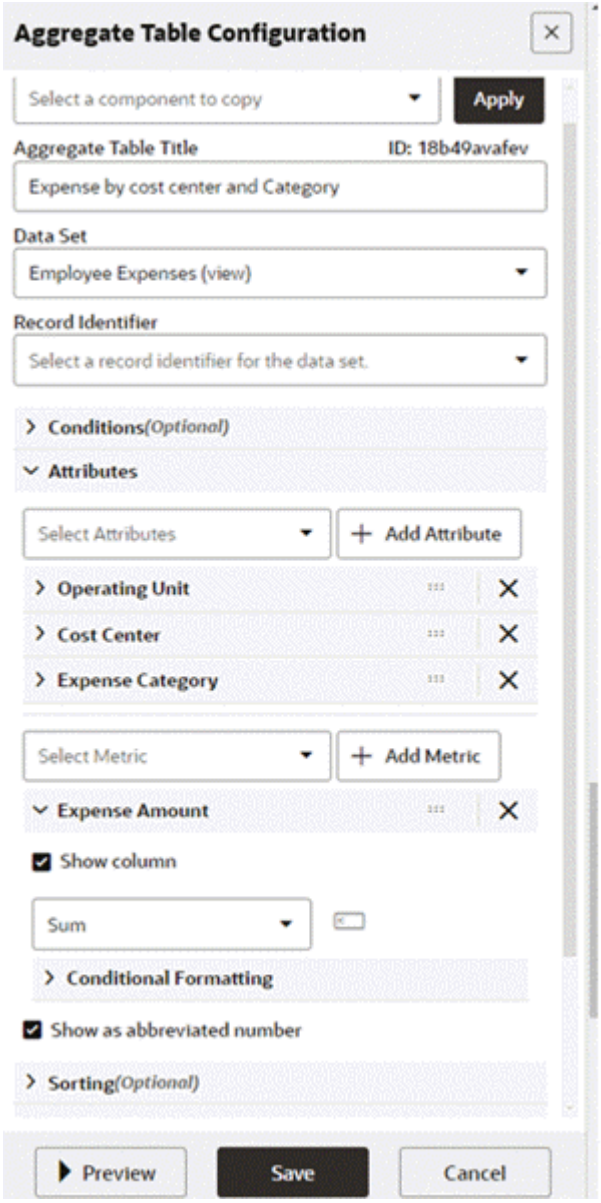
1. Define the title and the data set.

Defining the Title and Data Set for an Aggregated Table



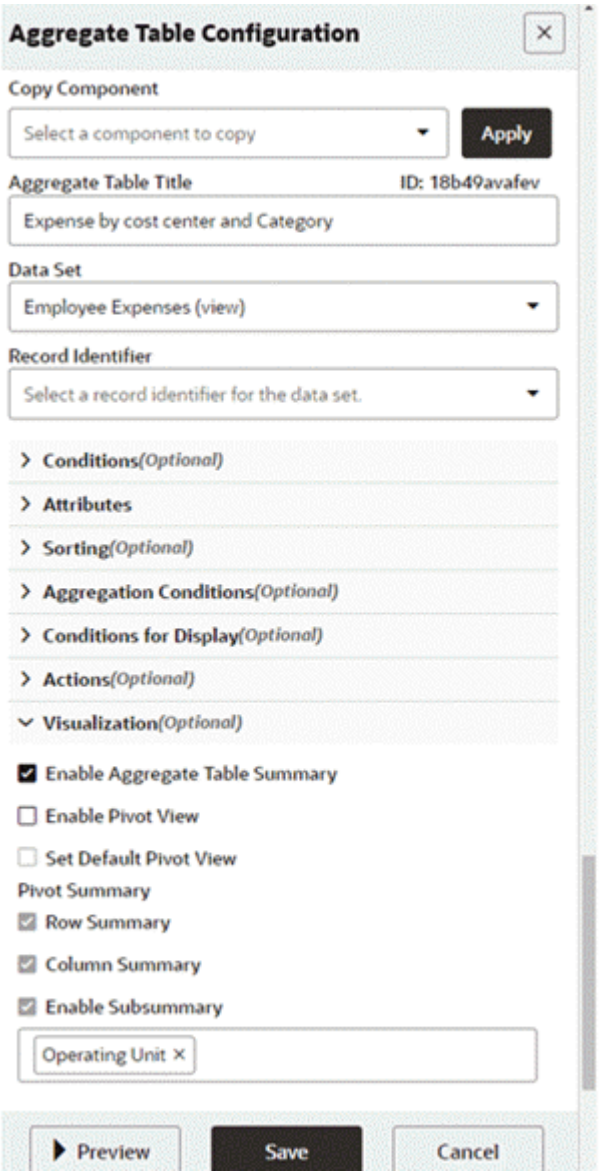
2. Add the attributes and metrics. The checkbox **Show as abbreviated number** is checked by default.

Configuring Attributes and Metrics for an Aggregated Table



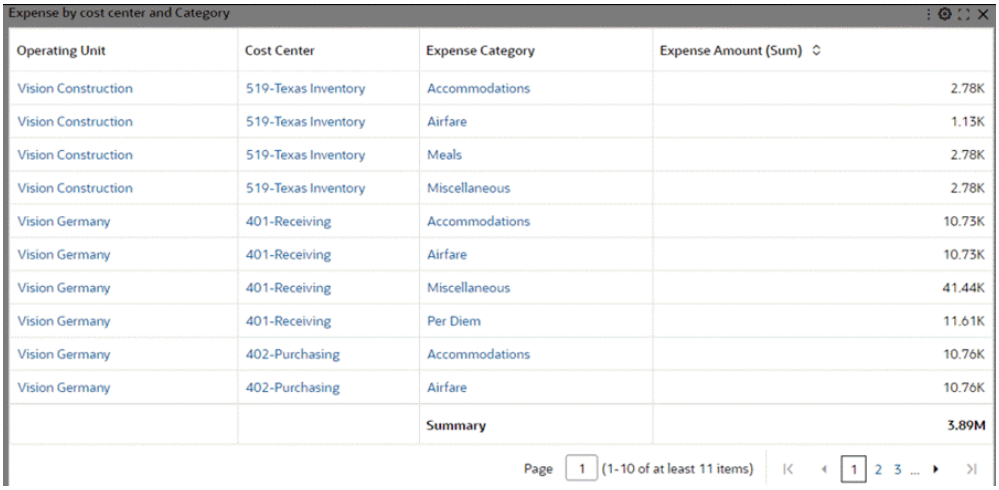
3. Enable the aggregate table summary by selecting its checkbox.

Enabling the Aggregate Table Summary



4. Preview the aggregated table.

Preview of an Aggregated Table



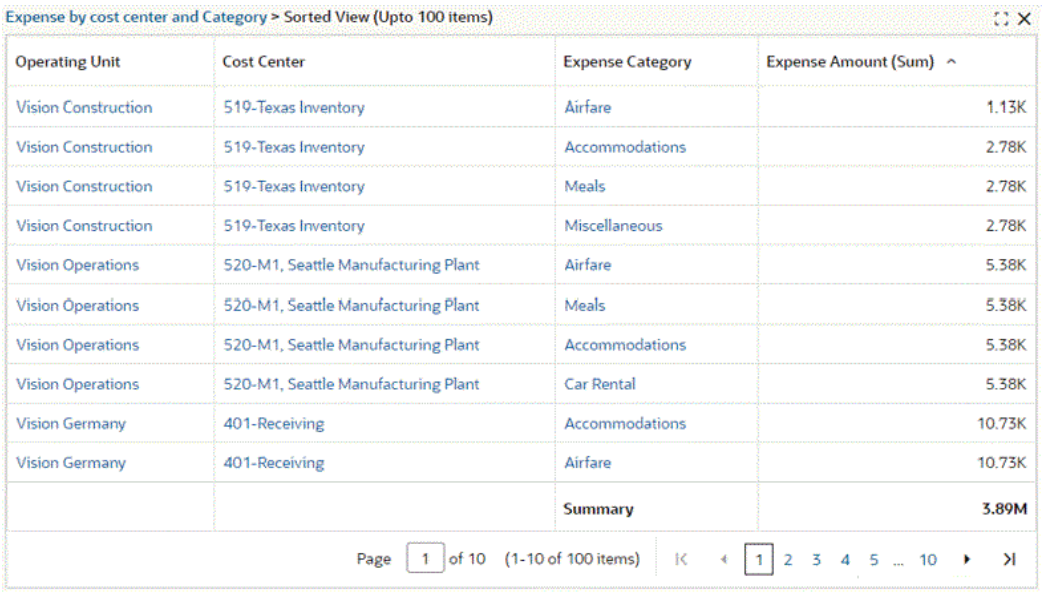
Expense by cost center and Category

Operating Unit	Cost Center	Expense Category	Expense Amount (Sum)
Vision Construction	519-Texas Inventory	Accommodations	2.78K
Vision Construction	519-Texas Inventory	Airfare	1.13K
Vision Construction	519-Texas Inventory	Meals	2.78K
Vision Construction	519-Texas Inventory	Miscellaneous	2.78K
Vision Germany	401-Receiving	Accommodations	10.73K
Vision Germany	401-Receiving	Airfare	10.73K
Vision Germany	401-Receiving	Miscellaneous	41.44K
Vision Germany	401-Receiving	Per Diem	11.61K
Vision Germany	402-Purchasing	Accommodations	10.76K
Vision Germany	402-Purchasing	Airfare	10.76K
		Summary	**3.89M**

Page 1 (1-10 of at least 11 items) 1 2 3 ...

Enhancements to aggregated tables in V12 allow users to sort by a metric. The sorted view is presented in an inline view, and it shows only shows up to 100 records after sorting.

Example of an Aggregated Table Sorted by Expense Amount (Sum)



Expense by cost center and Category > Sorted View (Upto 100 items)

Operating Unit	Cost Center	Expense Category	Expense Amount (Sum)
Vision Construction	519-Texas Inventory	Airfare	1.13K
Vision Construction	519-Texas Inventory	Accommodations	2.78K
Vision Construction	519-Texas Inventory	Meals	2.78K
Vision Construction	519-Texas Inventory	Miscellaneous	2.78K
Vision Operations	520-M1, Seattle Manufacturing Plant	Airfare	5.38K
Vision Operations	520-M1, Seattle Manufacturing Plant	Meals	5.38K
Vision Operations	520-M1, Seattle Manufacturing Plant	Accommodations	5.38K
Vision Operations	520-M1, Seattle Manufacturing Plant	Car Rental	5.38K
Vision Germany	401-Receiving	Accommodations	10.73K
Vision Germany	401-Receiving	Airfare	10.73K
		Summary	**3.89M**

Page 1 of 10 (1-10 of 100 items) 1 2 3 4 5 ... 10

Pivot View

Introduced in V8, the Pivot View feature presents an alternative visualization to aggregated table. A pivot view can be configured to include a grand summary row and

grand summary columns.

Example of a Pivot View

Quarterly Margin Summary

		Vision Operations			
Fiscal Year	**Fiscal Quarter Number**	Sales Revenue (Sum)	Cost of Goods Sold...	Gross Margin (Sum)	Gross Margin Perc...
2022	Q1	396.00	360.00	36.00	9.09
2021	Q4	109.89	0.00	109.89	100
2020	Q3	1.91K	0.00	1.91K	100
	Q1	0.00	20.00	-20.00	-200.00M
2011	Q3	-1.70K	-1.91K	214.30	-12.61
2010	Q1	-91.99K	-45.26K	-46.73K	50.8
2008	Q1	0.00	0.00	0.00	0
2007	Q2	8.00K	0.00	8.00K	100
	Q1	38.85K	19.40K	19.45K	50.07
	Summary	533.85M	398.47M	135.38M	25.36

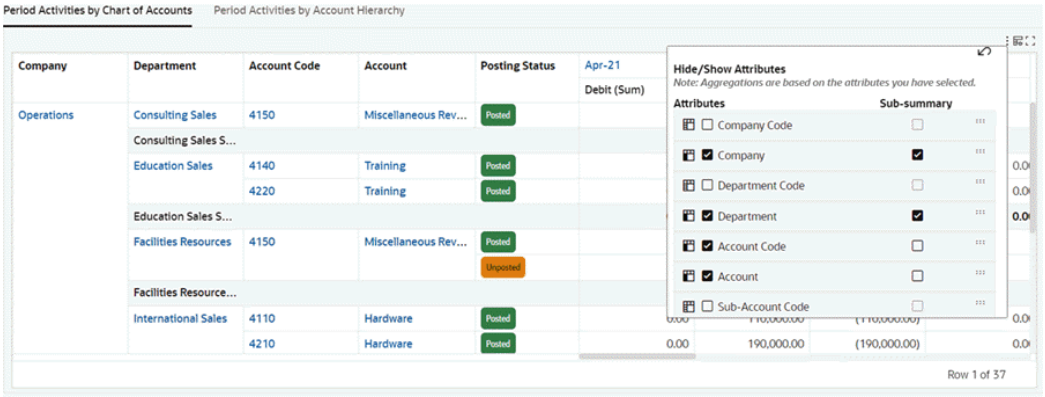
In addition to viewing the Pivot View data, users can change the pivot layout by show/hide attributes and metrics using runtime options.

Introduced in ECC V9, enhancements to both the Aggregated Table component and Pivot View now support the reordering of attributes and metrics from runtime. This capability allows users to pivot an aggregate table on any attribute, enabling them to view aggregations by setting any attribute as the column attribute. Additionally, in runtime options for Pivot view, to help users in the identification of attribute types of unique icons are displayed. Row attributes, column attributes and unselected attributes have distinct icons.

Introduced in ECC V10, enhancements to the Pivot view now offer sub-summary support for attributes outlined in the configuration. Additionally, users can conveniently enable or disable pivot sub-summary during runtime.

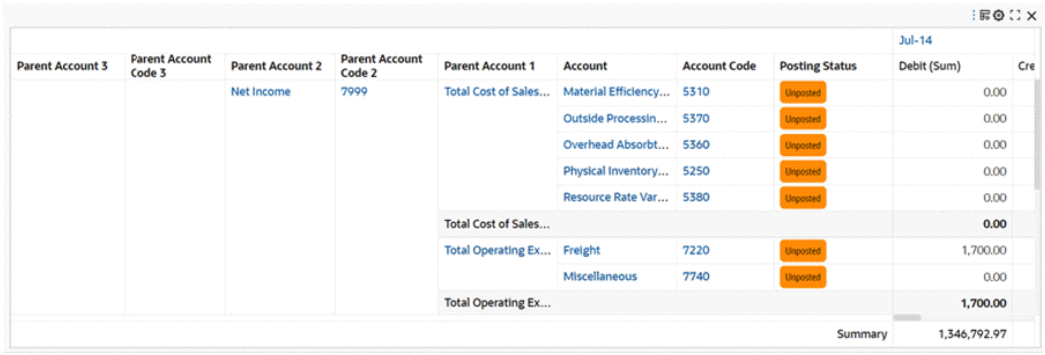
In V11, the runtime option is further enhanced, allowing end users to enable or disable Sub-Summary based on their preferred attribute.

Example of Pivot View Runtime Options



The Pivot View component allows for sorting by dimension (from A to Z or Z to A) or grand summary (from smallest to largest or largest to smallest).

Example of Pivot with Grand Summary and Sub-Summary



Note: Metadata-level color pinning is extended to the Aggregate Table component and corresponding Pivot View in V10. This feature is used in conditional formatting display. For example, an attribute such as "Posting Status" could have conditional formatting enabled from the metadata.

Configuration Options Specific to the Pivot View

Configuration options for Pivot View only are listed in the table below.

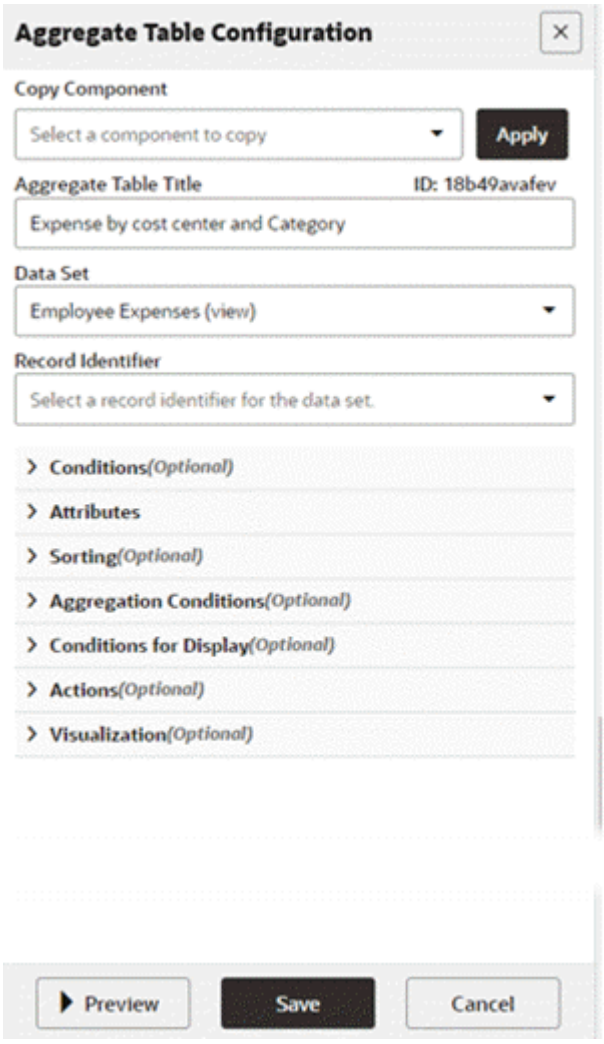
Configuration Options for Pivot View

Option	Description
Enable Pivot View	Enable Pivot View support. Allow switching between the aggregate table and the pivot view at runtime.
Set Default Pivot View	Set the pivot as the default view of the aggregated table.
Pivot Summary: Row Summary	Display a grand summary row.
Pivot Summary: Column Summary	Display grand summary columns.
Pivot Summary: Enable Subsummary	Display subsummary specific to the selected attributes.

Configuration Example

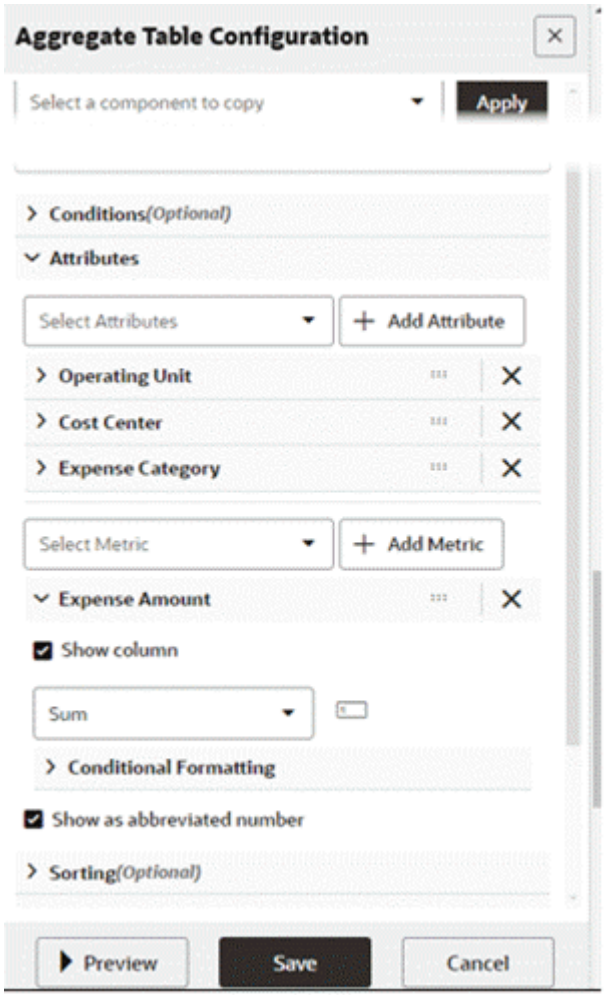
1. Define the title and data set.

Defining the Title and Data Set for Pivot View



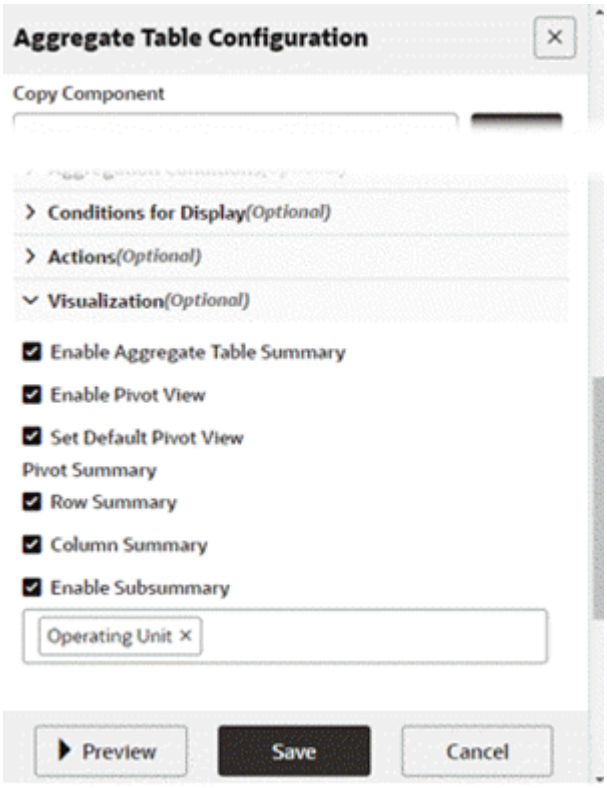
2. Add the attributes and metrics. **Show as abbreviated number** is checked by default.

Configuring Attributes and Metrics for Pivot View



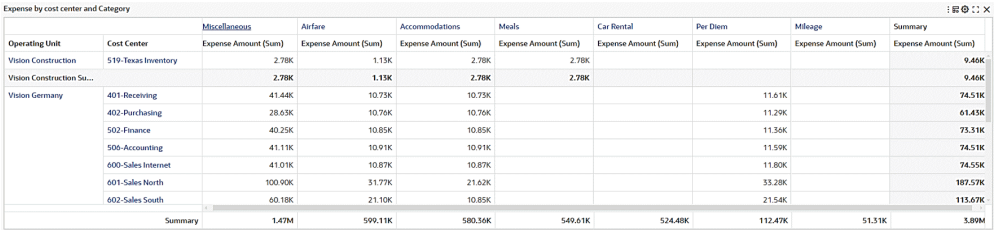
3. Enable Pivot View and its associated settings.

Configuring the Pivot View Settings



4. Preview the pivot view.

Preview of a Pivot View



Expense by cost center and Category

		Miscellaneous	Airfare	Accommodations	Meals	Car Rental	Per Diem	Mileage	Summary
Operating Unit	Cost Center	Expense Amount (Sum)	Expense Amount (Sum)	Expense Amount (Sum)	Expense Amount (Sum)	Expense Amount (Sum)	Expense Amount (Sum)	Expense Amount (Sum)	Expense Amount (Sum)
Vision Construction	519-Texas Inventory	2.78K	1.13K	2.78K	2.78K				9.46K
Vision Construction Su...		2.78K	1.13K	2.78K	2.78K				9.46K
Vision Germany	401-Receiving	41.44K	10.75K	10.75K			11.61K		74.51K
	402-Purchasing	28.63K	10.76K	10.76K			11.29K		61.45K
	502-Finance	40.25K	10.85K	10.85K			11.36K		73.31K
	506-Accounting	41.11K	10.91K	10.91K			11.59K		74.51K
	600-Sales Internet	41.01K	10.87K	10.87K			11.80K		74.55K
	601-Sales North	100.90K	31.77K	21.62K			33.28K		187.57K
	602-Sales South	60.18K	21.10K	10.85K			21.54K		113.67K
Summary		1.47M	599.11K	580.36K	549.61K	524.48K	112.47K	51.51K	3.89M

Dynamic Attribute Label in Pivot View and Available Refinements

Beginning with V9, designers can establish precedence rules in metadata, and these rules subsequently affect the behavior of the pivot view. Users can assign a label to a target attribute based on a predefined value of a trigger attribute, and dynamically, the corresponding defined label will be shown in the pivot view.

Note: This feature is supported in only the Pivot View component, and not in the Aggregate Table component.

Skipping Null Attributes in the Aggregate Table Component and Pivot View

Beginning with V9, the "skip null" feature is extended to the Aggregate Table component and Pivot view.

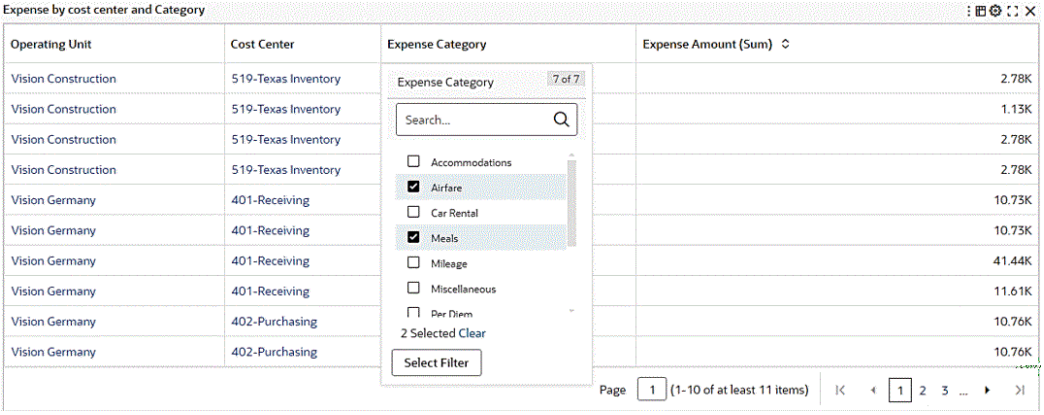
Designers can create an Aggregate table component or Pivot view and based on the presence of values in the attributes, the respective attributes will be displayed. That is, if an attribute has null values only, then the attribute will not be displayed. Any attribute that at least has one non-null value will be displayed.

Local Filters in Aggregate Table

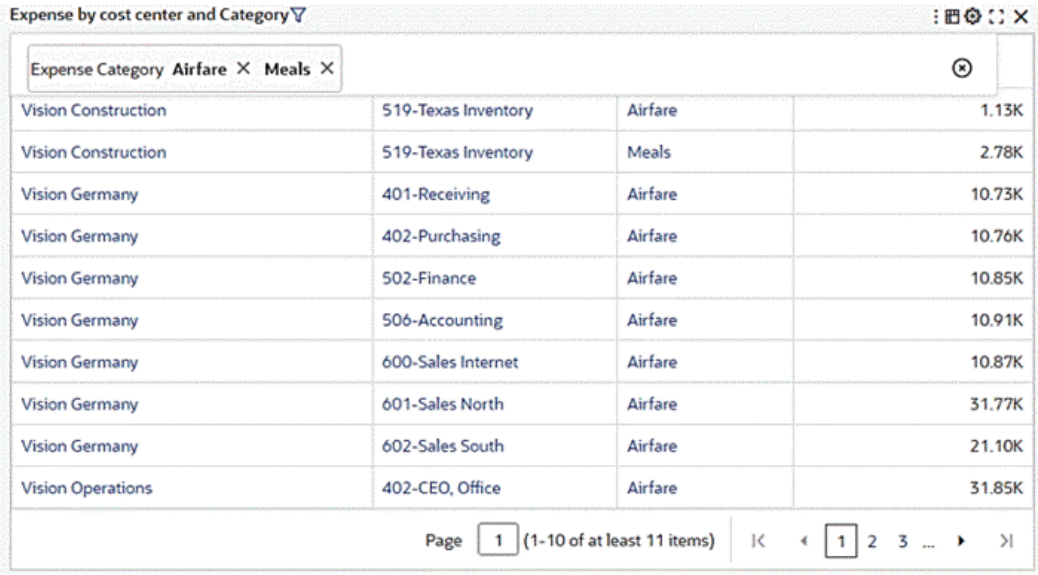
Introduced in ECC V12, local filters for aggregated table empowers users to focus only the records of interest without affecting the global context.

For example, in an aggregated table displaying records for expenses, a user could apply a local filter for only those expenses in the categories of "Airfare" and "Meals." With those filters applied, only the expenses in those categories are displayed.

Example of Applying Local Filters to an Aggregated Table



Example of an Aggregated Table with Local Filters Applied



Detailed Insight Components

Oracle Enterprise Command Centers offer detailed insights into the data though results tables and grids. This allows users to take the required action to resolve a process bottleneck, address an exception, or progress a business transaction.

Results Table

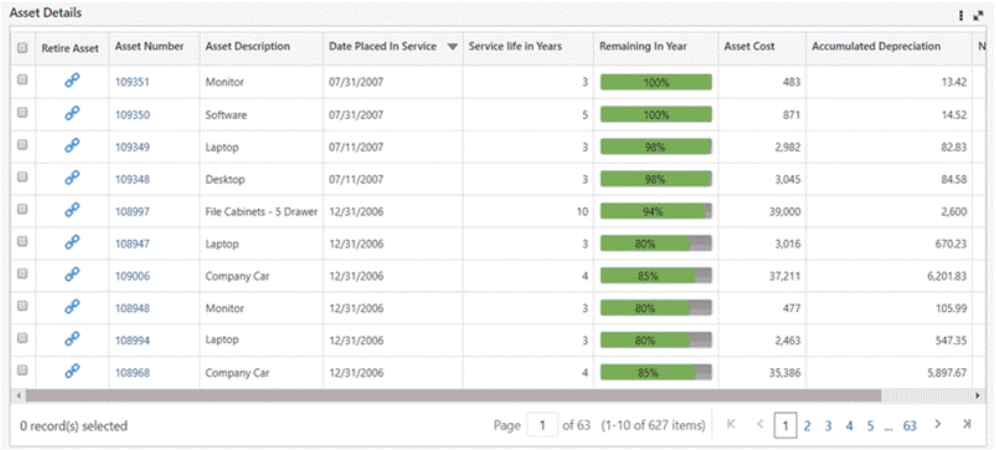
The Results Table component displays a set of data in a table format. A results table displays ten records in each page and when expanded at runtime, the results table displays up to 50 records per page.

Descriptive flexfield attributes can be configured as an attribute group in a results table.

The data displayed in the Results Table component is either:

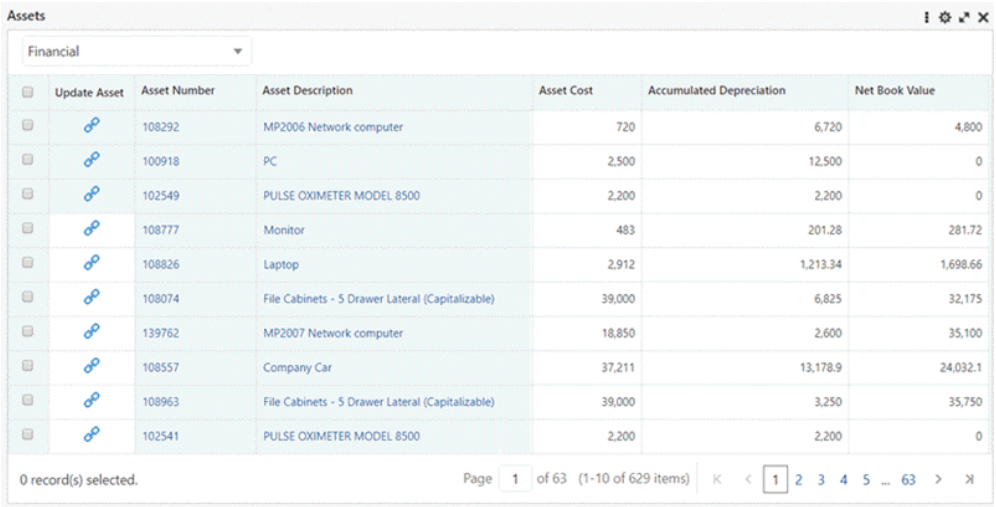
- A flat list of records from a selected data set. Each row represents a single record. The columns contain attribute values for that record.

Results Table - Flat List



- A grouped list of attributes. Each group represents a functional or logical grouping for a set of attributes.

Results Table - Attribute Groups



The Results Table component supports several new features in V10, including: conditional formatting, freezing of columns, grouping of row actions under one icon, flexibility in selecting rows, and end user personalization capabilities, These features are described below.

The Results Table component supports conditional formatting. For categorical attributes, the conditional formatting is driven by metadata color pinning. For numerical attributes, the conditional formatting is supported from the component's configuration.

The Results Table component supports freezing of columns:

- Default Freezing: Row selection checkbox column, Displayed detail columns are frozen by default. In the case of attribute groups, the persistent attributes are frozen by default, but the designer has the flexibility to alter this in the configuration.

- User-Defined Freezing: Every attribute in the results table has a configuration allowing for freezing. In the case of persistent attributes, the freezing is enabled by default.

The Results Table component displays all the row actions grouped under the horizontal overflow icon (…). This column is also frozen by default. No such grouping exists for result tables having only one row action.

The Result Table offers flexibility in selecting rows with the following options:

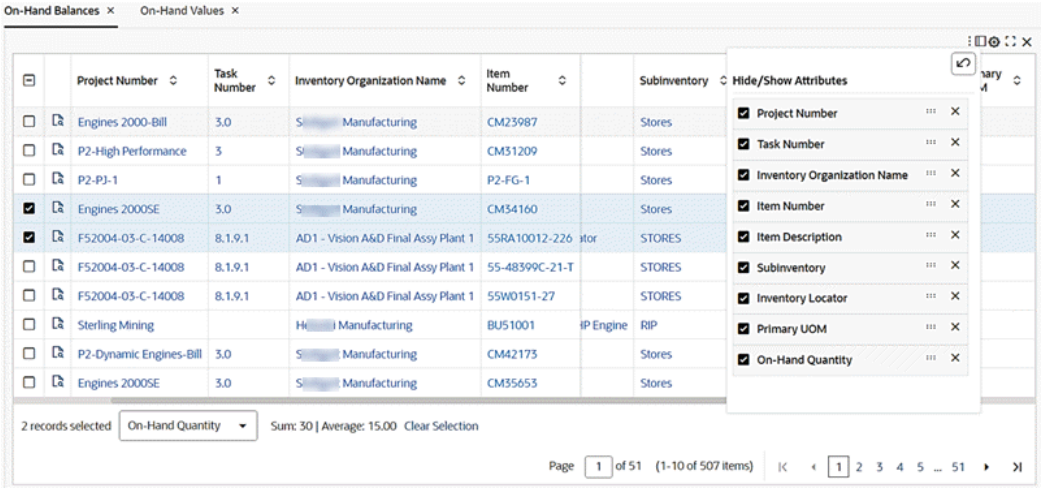
- Multiple: Multiple rows can be selected. This is the default option.

- Single: Only one row can be selected a time.

- Disable: Row selection is disabled. This option is useful in scenarios where the results table appears on a single page and there is no need to perform any action on the table.

Beginning in V10, the Results Table component allows end users to hide, show, reorder, and delete attributes in runtime. When multiple records are selected during runtime, the user can access summary statistics (sum and average) related to numerical attributes. This information appears in the footer of the results table. Both of these features are considered in the scope of End User Personalization and are preserved for a user from session to session.

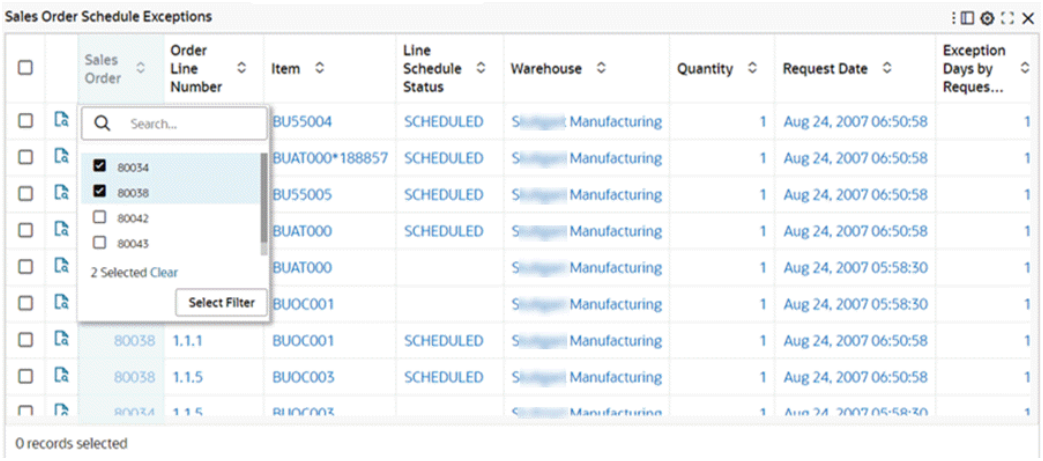
Hide/Show Attributes Window for a Results Table



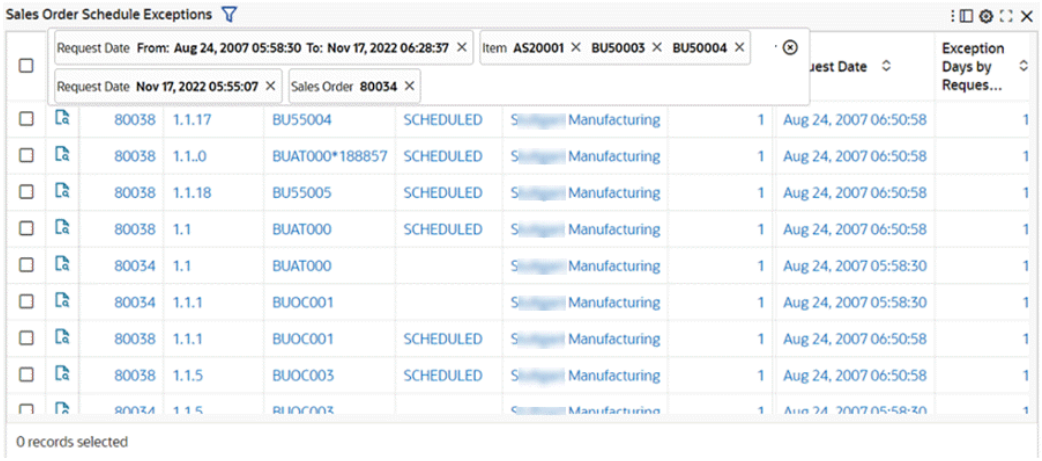
Beginning with V11, a local filter is supported in a results table to focus only the records of interest without affecting the global context. Except for calculated attributes, all attributes support local filtering at runtime. Local filtering is also applicable for the inline results tables.

A user can hover over a column title to view and select local filters.

Example of a Local Filter for a Column in a Results Table

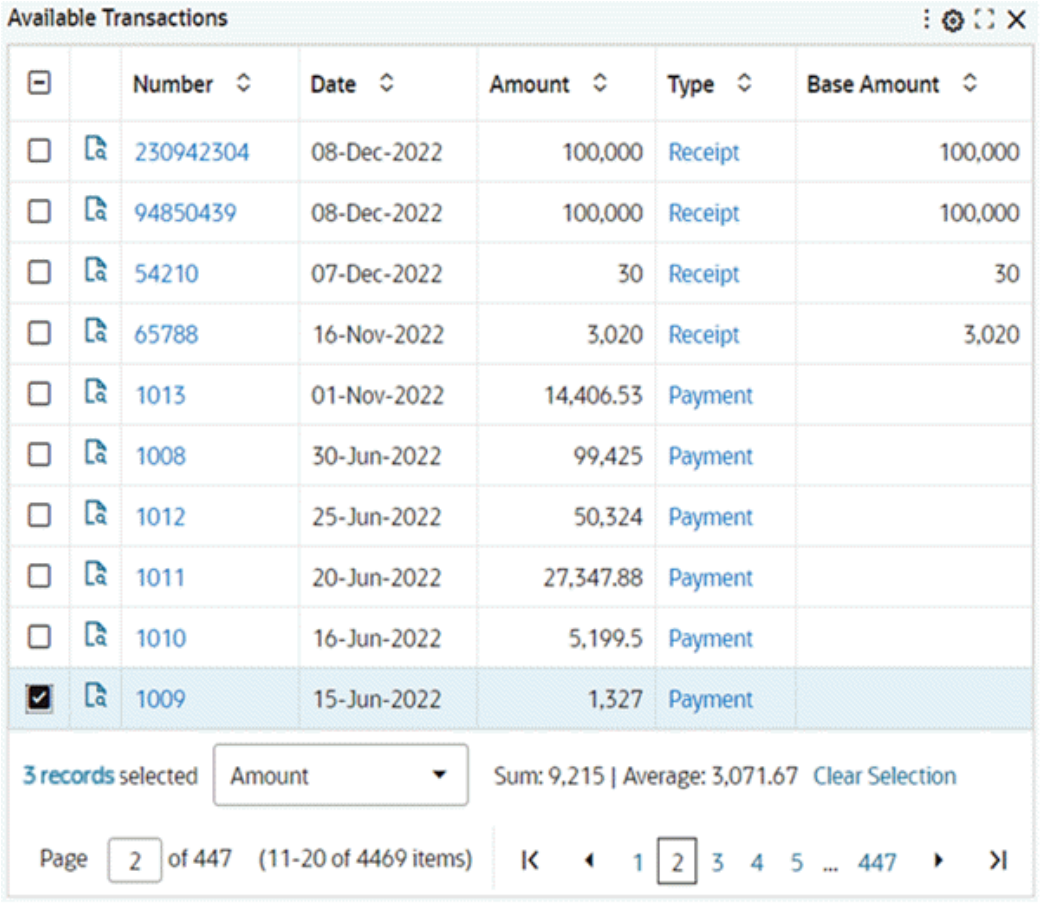


Example of a Results Table with Local Filters Selected



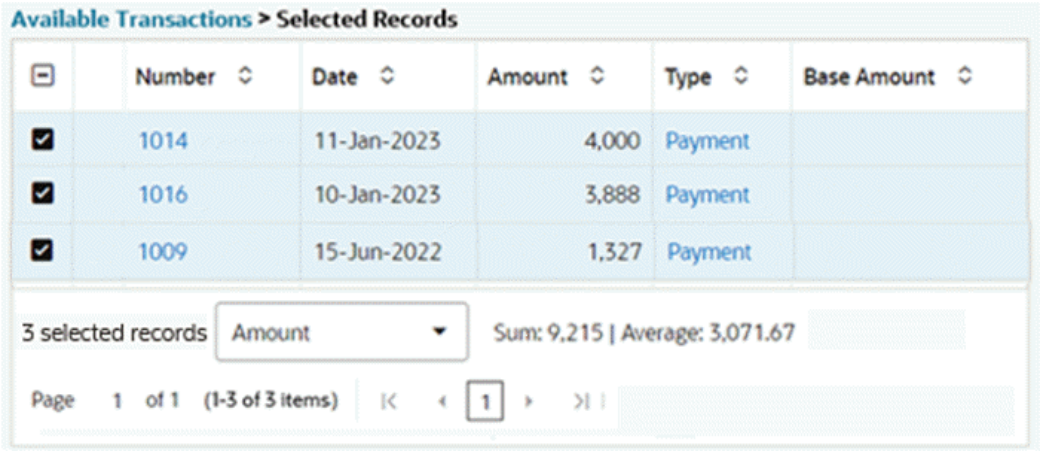
Introduced in ECC V11, the Show Selected Records view is supported in results tables. The footer of the result table is enhanced to show a hyperlink when records are selected. Upon clicking the hyperlink, a user can access all the selected records under a single view. The selected records are overlaid on the original table with a breadcrumb on the table title so the user can go back to the original table.

Example of a Results Table with Three Records Selected across Pages and Hyperlink in Footer



Upon clicking the hyperlink, a user can access all the selected records under a single view. The selected records are then overlaid on the original table with a breadcrumb in the table title to go back to the original table.

Example of Selected Records in a Results Table



Configuration for a Results Table

Configuration options are listed below.

Common Configuration Options

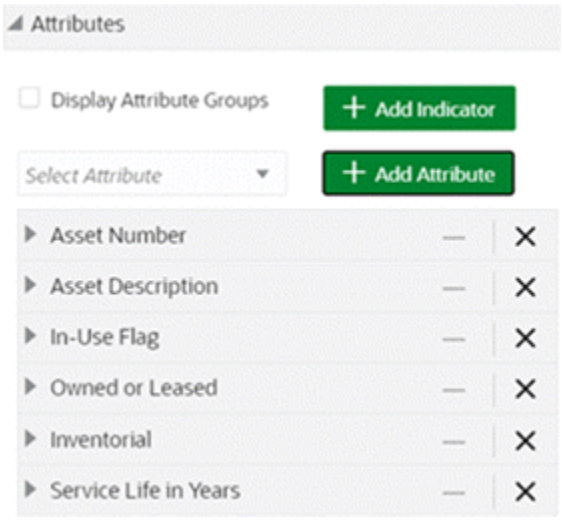
Option	Description
Title	Component title.
Data Set	Select one data set.

Flat List Configuration

Flat List Configuration Options

Option	Description
Attributes	Select an attribute and click **+ Add Attribute**.
Indicators	Click **+ Add Indicator** to add an Indicator.

Results Table - Flat List Configuration



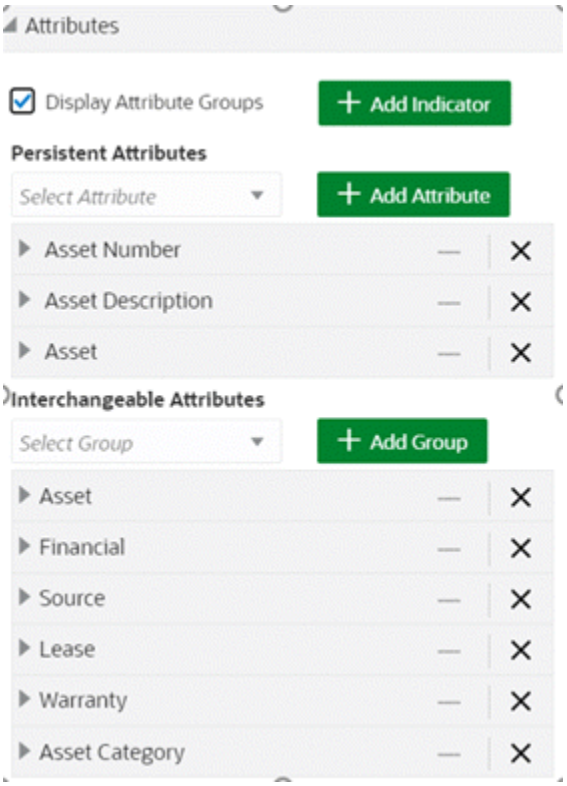
Attribute Group and Attribute Level Configuration

Attribute Group Configuration Options

Option	Description
Display Attribute Group	Select **Display Attribute Group** to allow the display of attribute groups in the results table.
Persistent Attributes	Click **+ Add Attribute** in the persistent attribute section. • To set the display order of the attributes, drag each attribute to the appropriate location in the list. The column at the top of the list displays at the far left of the table. • To remove a column from the Persistent Attribute list, click its delete icon. • To add an indicator click **+ Add Indicator**.

Option	Description
Interchangeable Attributes	Click + Add Group in Interchangeable Attributes section. Add attribute groups: • Cannot change the name of an attribute group • Can re-order the attribute groups, and the attributes within a group

Results Table - Attribute Group Configuration



Attribute Level Configuration Options

Option	Description
Show Column Checkbox	To show or hide the indicator column in the result table.

Option	Description
Freeze Columns	To apply or remove freezing to a column. This option is enabled for persistent columns.
Conditional Formatting	To enable conditional formatting for numerical attributes based on the column value.

Indicator Configuration

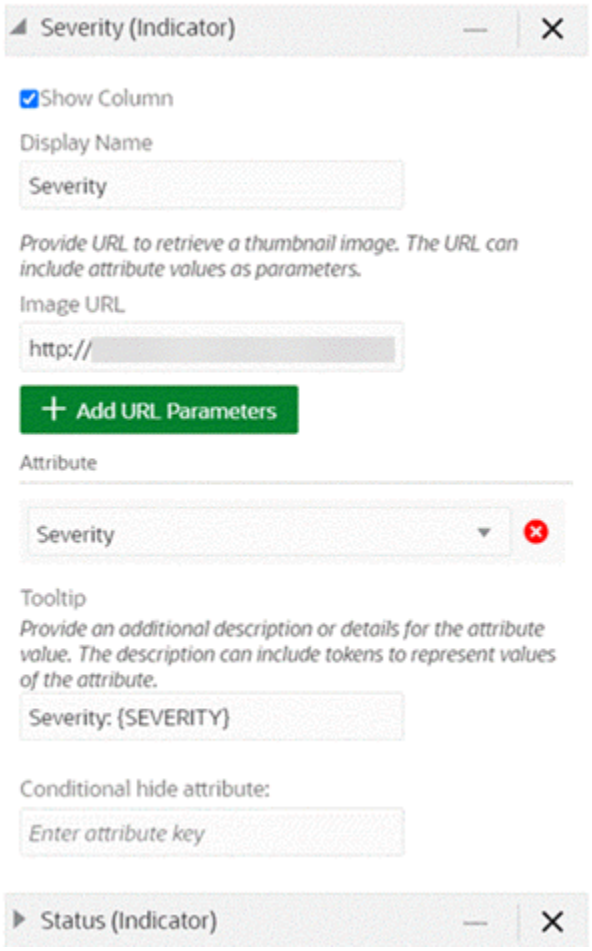
Beginning with V9, the Results Table component supports displaying indicators in order to highlight vital parameters with visual cues rather than text, so that a user can quickly spot them. In an 'attribute group' setup, indicators can only be added in the section of persistent attributes.

Indicator Configuration Options

Option	Description
Show Column Checkbox	This box is used to show or hide the Indicator Column in the results table.
Display Name Text Box	The name of the indicator. The same name is used as a column name for the indicator.
Image URL Text Box	The URL to retrieve an image as a thumbnail. The URL can include attribute values as parameters
Add URL Parameter Button	The **+ Add URL Parameters** option defines the attribute passed as a parameter to dynamically retrieve the indicator.
Tooltip Text Box	This box is used to provide additional details for the attribute value. The description can include tokens to represent the value of the attribute.

Option	Description
Conditional Hide Attribute Text Box	This box provides a Boolean attribute based on which indicators can be shown/hidden dynamically at runtime.

Results Table Indicator Configuration



Specific Configurations

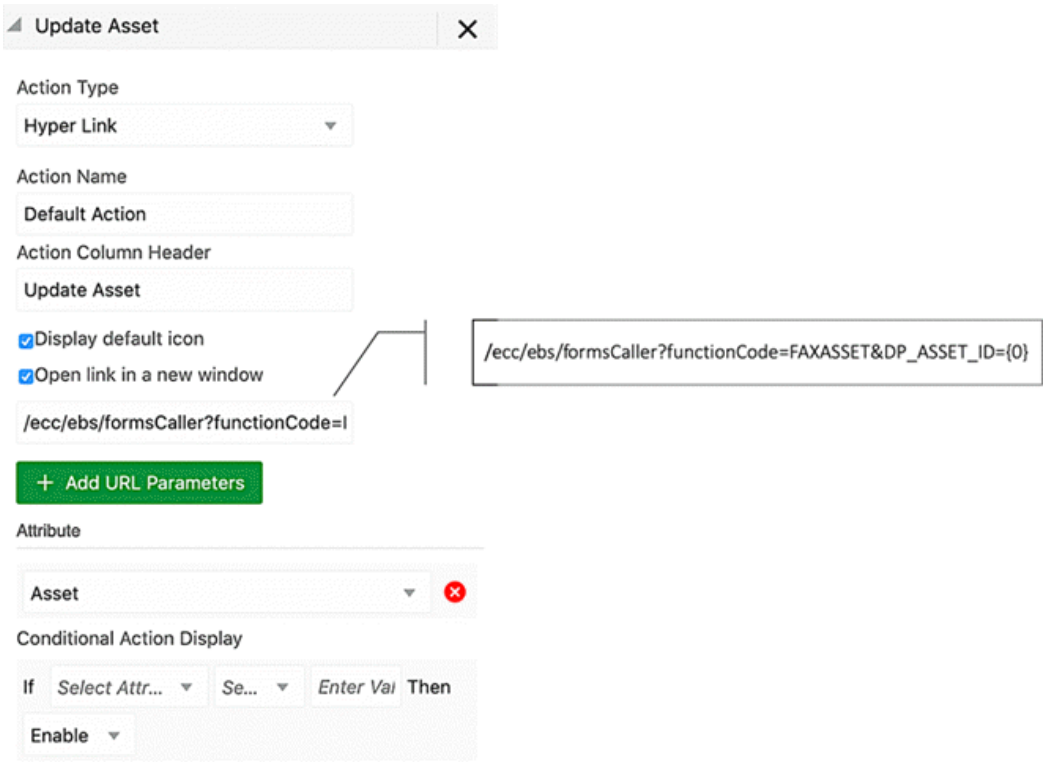
Actions - Row Action

Select **+Add Action** and select Action Type: Hyperlink.

- In the **Action Name** field, type a name for the action.
- To display the hyperlink as an icon, check the **Display Default Icon** box.

- To display the hyperlink in a separate browser window, check the **Open link in a new window** box.
- In the **URL** field, type the URL to link to.
- To add attribute values to the URL:
  - Click **Add URL Parameters**.
  - Select the attribute name.
- To add a conditional action display, select the following:
  - Attribute
  - Operator
  - Value
  - Enable or disable the action

Row Action Configuration

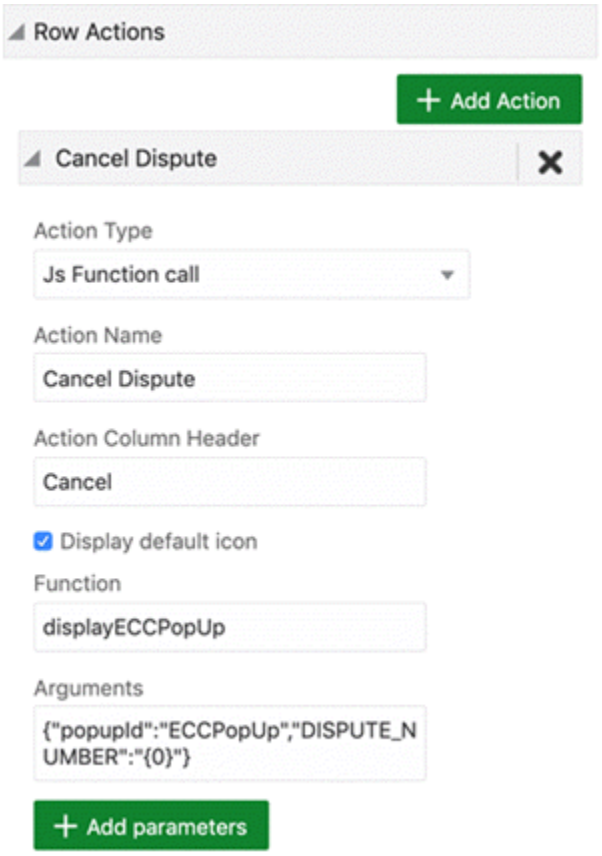


Actions - Row Action

Select **+Add Action** and Action Type: Js Function Call.

- In the **Action Name** field, type a name for the action. This will serve as a tooltip.
- In the **Function** field, type the function name.
- To add arguments to the function:
  - Click **Add Parameters**.
  - Select attribute name(s).
- To add a conditional action display, select the following:
  - Attribute
  - Operator
  - Value
  - Enable/disable the action

Row Action (Js Function Call) Configuration



Actions - Action Menu

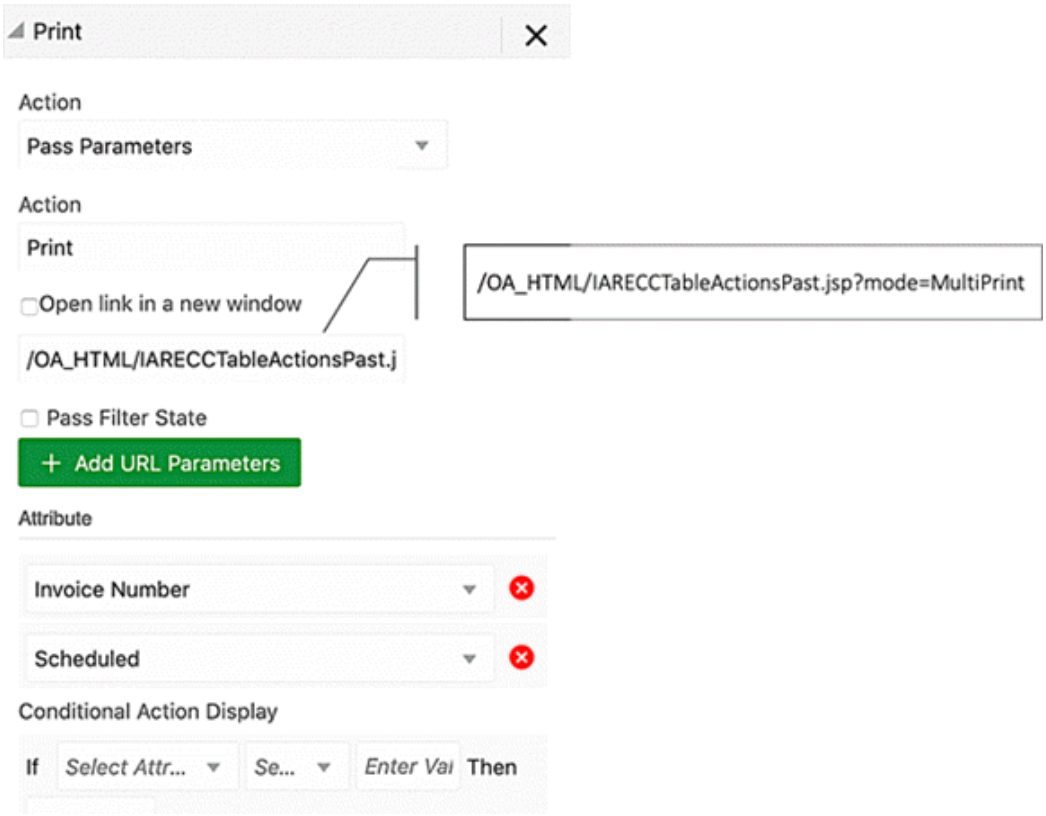
Select **+ Add Action** and select Action Type: Pass Parameters

- In the **Action Name** field, type a name for the action.
- To display the hyperlink in a separate browser window, check the **Open link in a new window** box.
- In the **URL** field, type the URL to link to.
- To allow applying a bulk action on all the records:
  - Select 'Pass Filter State'.
  - Note: The user will be warned if the results table contains more than the supported number of records.
- To add attribute values to the URL:
  - Click **Add URL Parameters**.

- Select the attribute name.

- To add a conditional action display, select the following:
  - Attribute
  - Operator
  - Value
  - Enable/disable the action

Results Table Action Menu Configuration



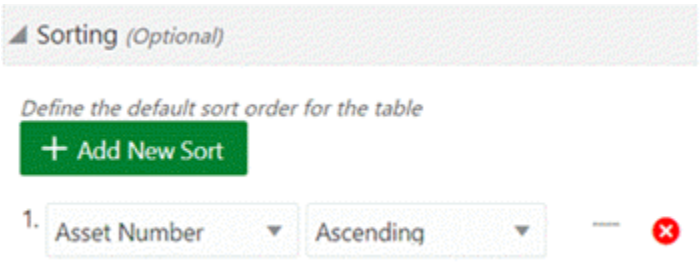
Sorting

Sorting allows a user to configure the default sort order to use for the table. To add a new sort rule:

- Click **+ Add Sort**.

- Select an attribute and the Ascending or Descending sorting option.

Results Table Sorting Configuration



Visualization Configuration

Visualization configuration is optional. There are three options for Record Selection Mode.

Record Selection Mode Options

Options	Description
Multiple	The default option. Multiple rows can be selected at a time.
Single	Only one record can be selected at a time.
Disable	Row selection is disabled. This option is useful where the results table is on a single page and there is no need to perform any action on the table.

Compare Configuration

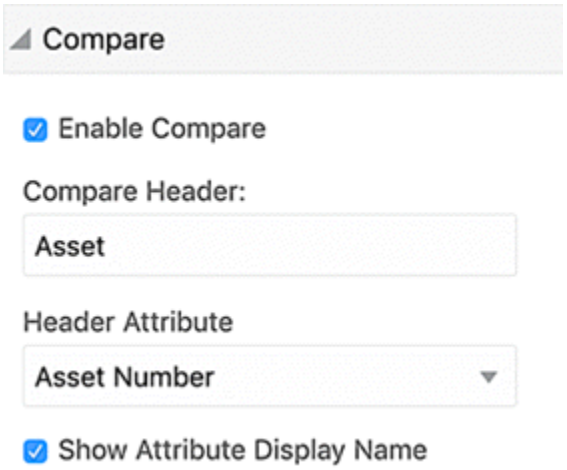
The Results Table component includes a Compare option on the Actions menu for comparing selected items. The configuration options for the Compare option are described below.

Configuration Options

Options	Description
Enable Compare	Checkbox to enable compare at runtime
Compare Header	Allow adding static test as a compare header

Options	Description
Header Attribute	Display compare header as a tokenized attribute
Show Attribute Display Name	Flag to display both attribute label and value

Compare Configuration



Configuration Matrix

The options to display the column header are described in the following table.

Configuration Matrix

Header	Attribute Header	Display	Example
Empty	Empty	Record n	Record 1
Text	Empty	Text n	Item 1
Text	Attribute Name	Text: Attribute Value	Item: LCD Thin Panel Monitor
Empty	Attribute Name	Attribute Display Name: Attribute Value	Description: LCD Thin Panel Monitor

The options to control the display of attribute groups in the Compare window are listed

in the following table.

Options for Display of Attribute Groups in the Compare Window

Option	Description
In Metadata Attribute Groups, Include in user actions	Checkbox to enable groups to be displayed in Compare window
In Results Table Configuration, Display Attribute Groups	Checkbox to enable groups to be displayed in the results table and Compare window (Groups displayed will be a union between the ones configured in metadata and in the results table)
In Results Grid Configuration, Use Attribute Groups	Checkbox to enable groups to be displayed in Compare window (Groups configured in metadata will be displayed)

The options to control the display of grid attributes in the Compare window header are listed in the following table.

Options for Display of Grid Attributes in the Compare Window Header

Option	Description
In Results Grid Cell Configuration, accessible in Compare and Record Details (Available for grid items: Text Input and Button)	Checkbox to display grid item in Compare window header

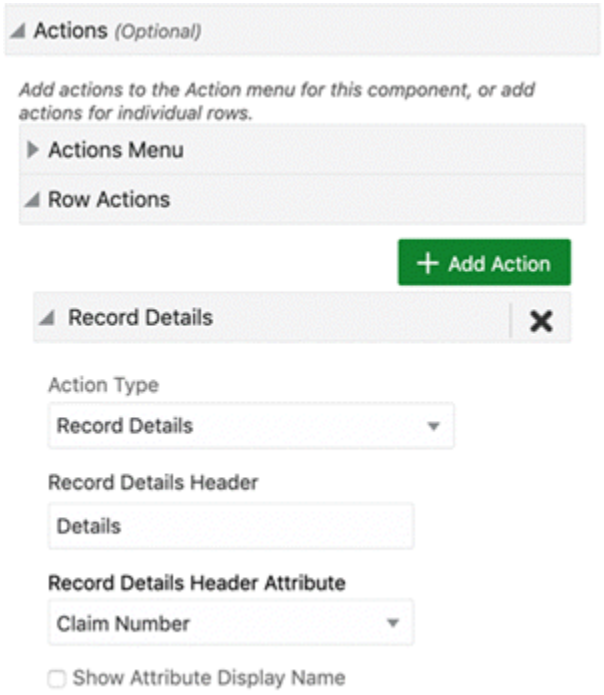
Record Details Configuration

The Record Details option allows you to view all the significant information of a record in a tabular format. The Record Details option displays the details in a window similar to the Compare option window . Attributes in a record details window can also be organized into respective groups, if configured. Similar to the Compare window, a Record Details window also supports dynamic titles and column header grid actions.

Configuration Options for Record Details

Options	Description
Add action	Row action type: Record details
Record Details Header	Allow adding static test as a compare header
Header Attribute	Display record details title as a tokenized attribute
Show Attribute Display Name	Flag to display both attribute label and value

Record Details Configuration for Results Table



The options to display the column header are described in the following table.

Configuration Matrix

Header	Attribute Header	Display	Example
Empty	Empty	Record n	Record 1
Text	Empty	Text n	Item 1
Text	Attribute Name	Text: Attribute Value	Item: LCD Thin Panel Monitor
Empty	Attribute Name	Attribute Display Name: Attribute Value	Description: LCD Thin Panel Monitor

The options to control the display of attribute groups are described in the following tables.

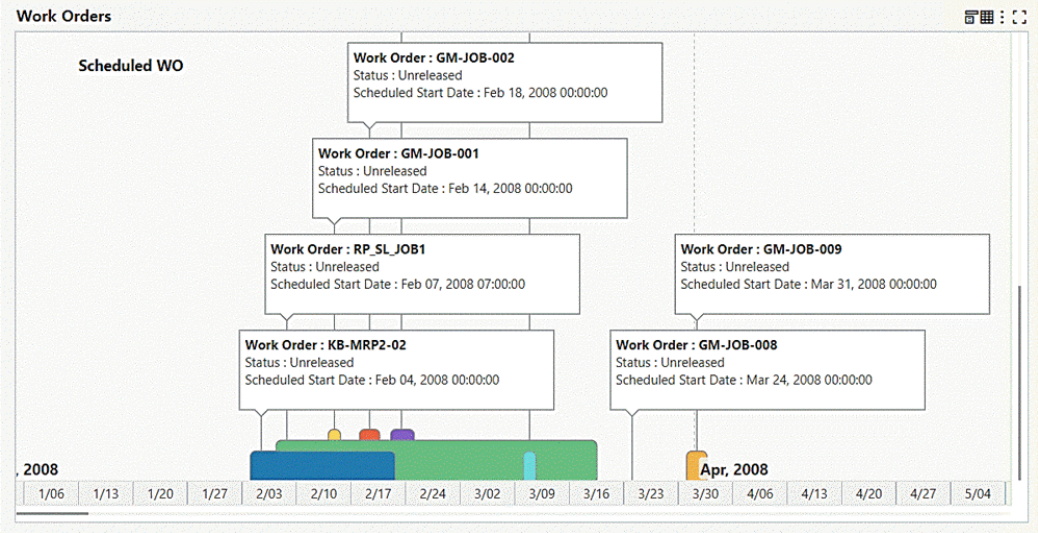
Options to Control Display of Attribute Groups in Record Details

Option	Description
In Metadata Attribute Groups, Include in user actions	checkbox to enable groups to be displayed in record details.
In Results Table Configuration, Display Attribute Groups	checkbox to enable groups to be displayed in results table and record details. (Groups displayed will be a union between the ones configured in metadata and in Results Table.)

Timeline View

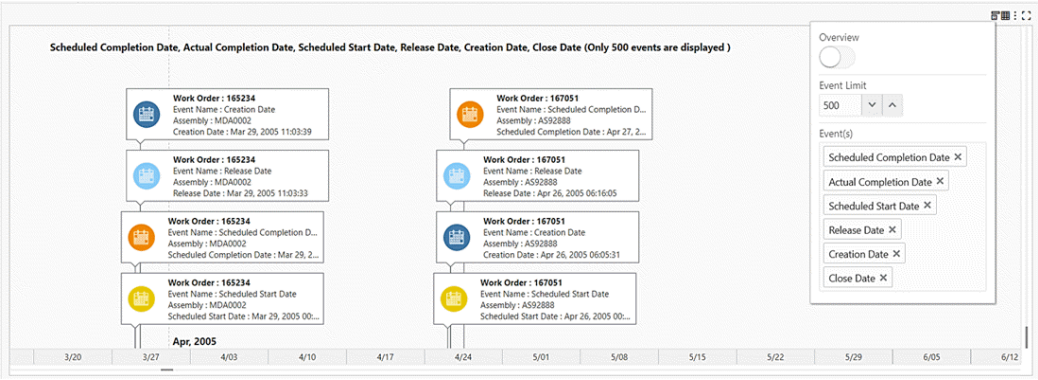
Introduced in ECC V8, Timeline view is supported in results table as an alternative visualization to the default results table. The timeline view is an interactive data representation of a time period, with key events marked along in chronological order. A user can quickly navigate forward and backward within a defined time range. Events are represented as timeline items, and each event can have a duration based on the start and end date of that event. A user can switch between the results table and the timeline view using the timeline icon.

Example of a Timeline View



The full range of available dates is based on the earliest and latest available dates for the date attributes displayed on the timeline. The timeline can have more than one event; a user can select which event to be displayed or display more than one event using the runtime options. Also, a user can control the event limit displayed on the timeline with the option to enable/disable the overview marque.

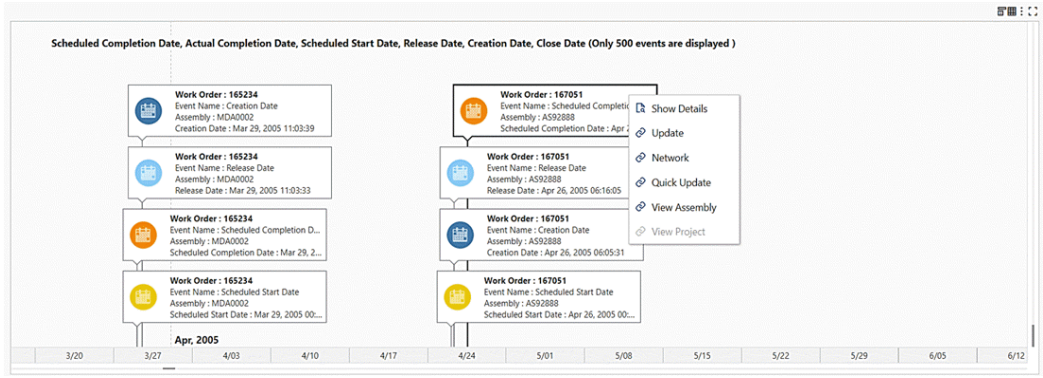
Example of Timeline Runtime Options



A user can view a set of actions from the timeline through the timeline context menu by right-clicking on the timeline event.

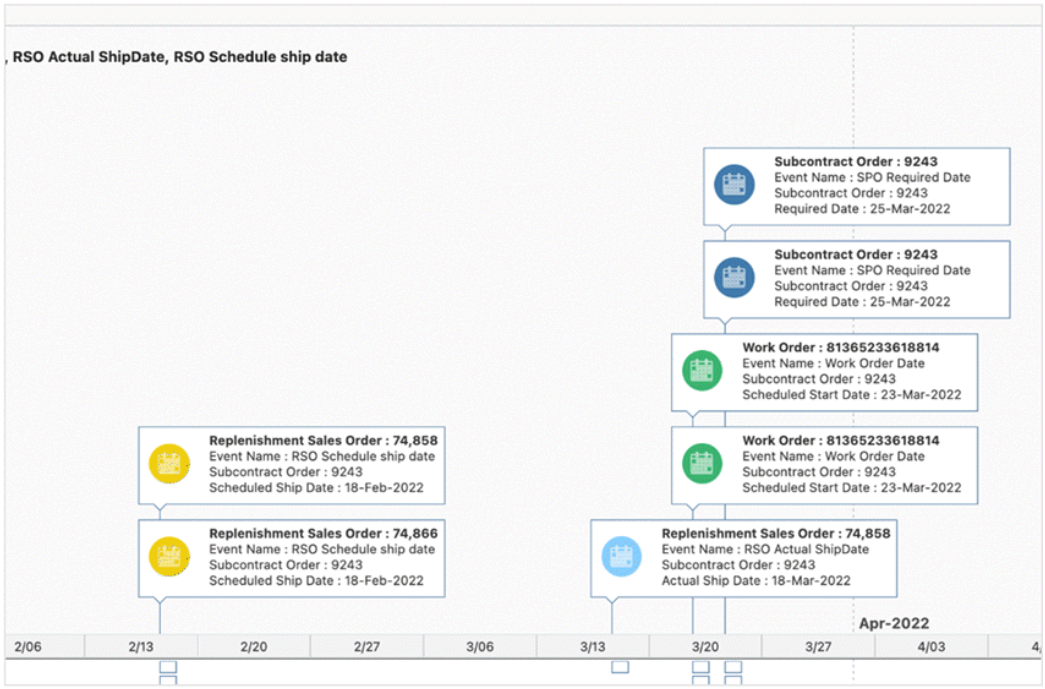
> **Note:** The timeline supports only row actions configured at the result table level.

Example of Timeline Runtime Options

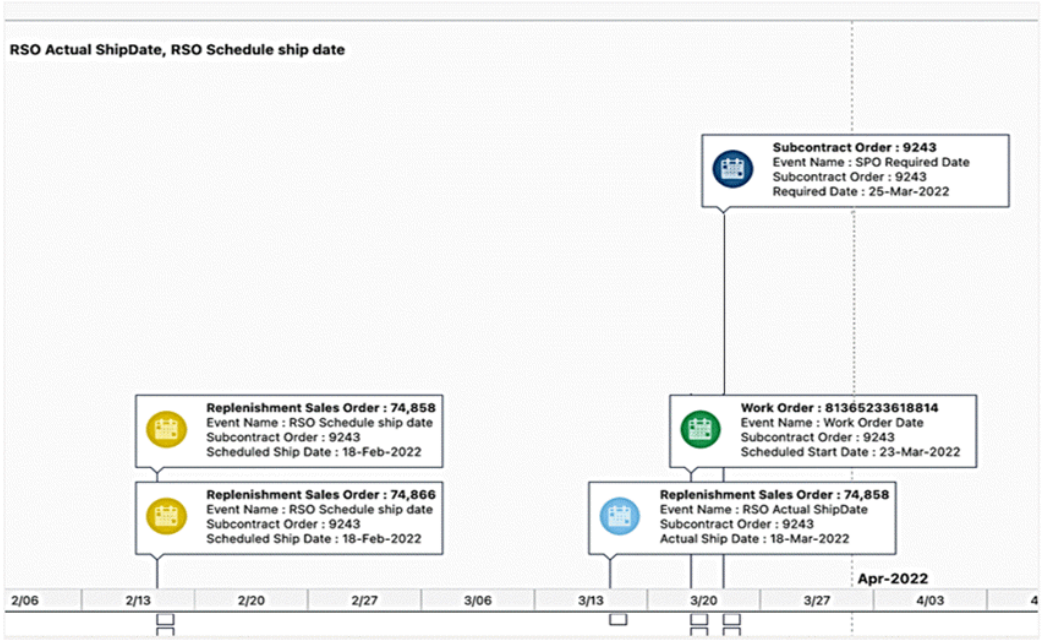


Beginning with V8, designers are empowered with the ability to regulate the granularity of events displayed on the timeline through an **event identifier**. This capability enables designers to show only unique events based on Title, Description 1, Description 2, Start Date, and End Date.

Example of Timeline Events Before Application of Event Identifier

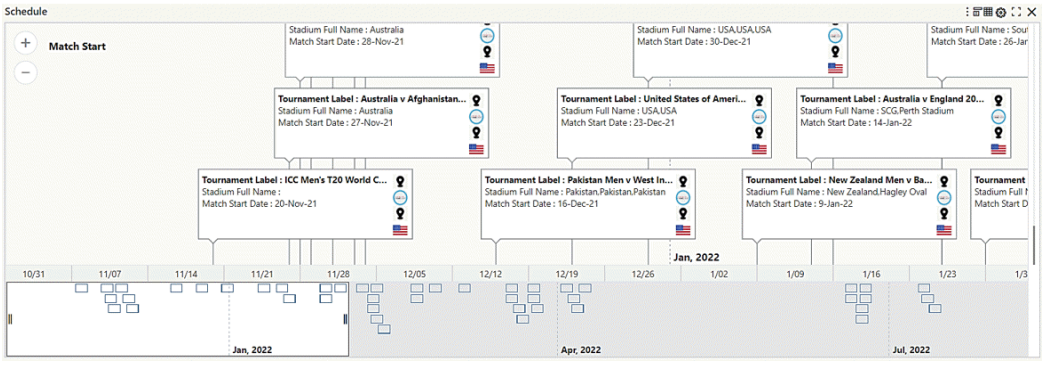


Example of Timeline Events after Application of Event Identifier



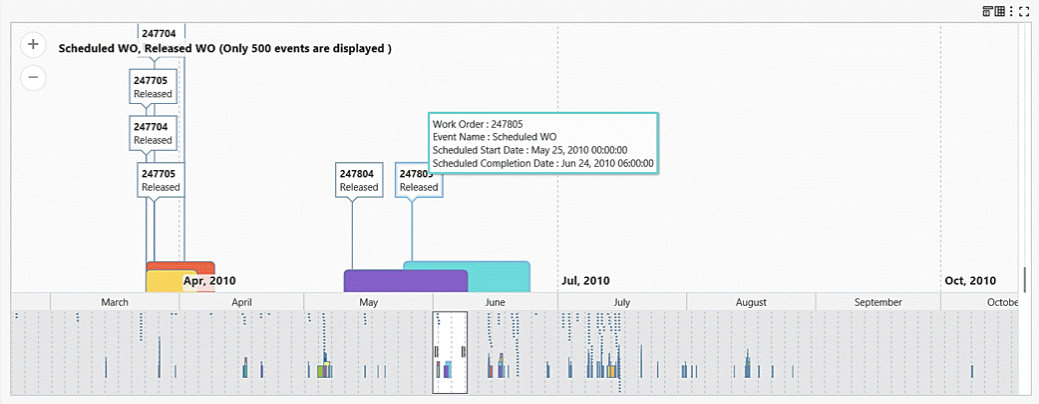
Beginning with V9, support for displaying indicators is added in timeline view. A timeline can display as many as four indicators during runtime. However, there is no restriction on a designer to specify a fixed number of indicators. Only the indicators configured in Results Table can be added in a timeline event. In addition to this, the timeline supports a configuration-driven tooltip to provide additional information for each timeline event. Only the attributes configured in Results Table can be added as a tooltip.

Example of Timeline View with Indicators



The timeline does not support images in Internet Explorer; the event title and description are only displayed in the event bubble, and full event details are displayed in the event tooltip.

Example of Timeline in Internet Explorer Browser



Specific Configuration Options for Timeline View

The following table lists general timeline configuration options.

General Timeline Configuration

Option	Description
Enable Timeline View	Box to enable timeline view.
Default View	Makes the timeline the default view of the results table.
Timeline Overview	Enables draggable timeline and overview to pan around the Viewport.
Event Limit	Sets the limit of the number of events displayed on the timeline. This number can be changed at runtime.
Component Height	Sets the timeline height in pixels.

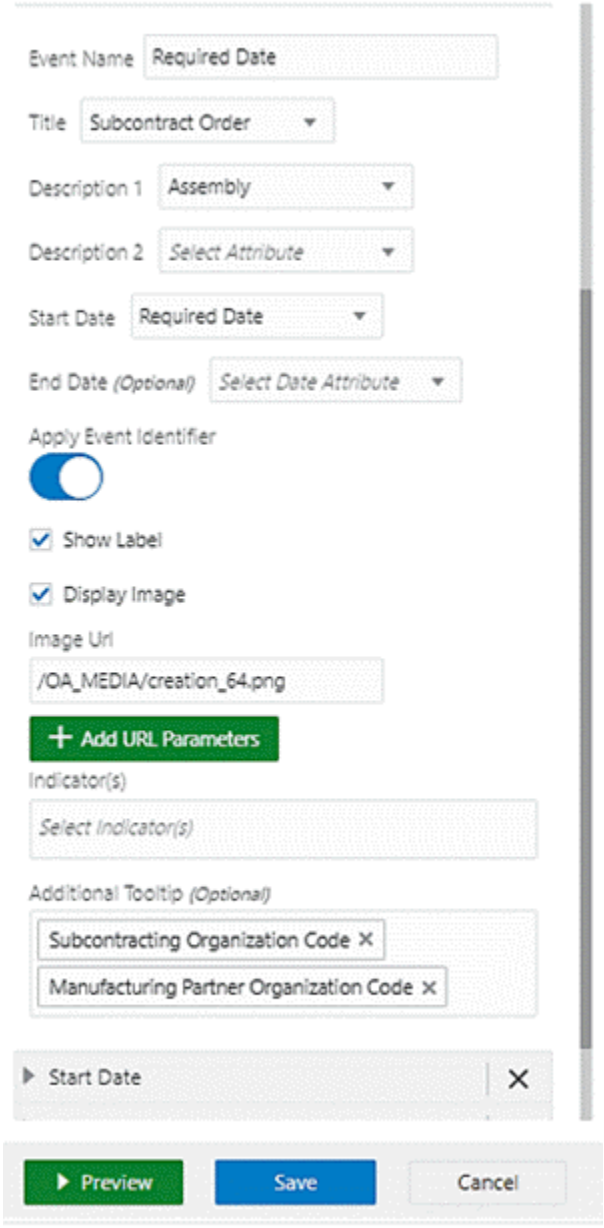
The following table lists configuration options for specific events.

Event Configuration

Option	Description
+ Add Event	Click on the **Event** button to add timeline events.
Event Name	Enter the event title that will be displayed on the timeline.
Title	Select the main attribute that will be timeline event.
Description 1	Specify a description attribute.
Description 2	Specify another description attribute.
Start Date	Specify the event start date.
End Date (optional)	Specify the event end date. If this option is configured, the timeline displays the event duration based on the start and end dates.
Apply Event Identifier	Enable this option to show only unique events based on Title, Description 1, Description 2, Start Date, and End Date.
Show Label	Use this box to show or hide the attribute label.
Display Image	Select this option to display an image in the timeline. Note that the image will be visible only in the case of multiple events being configured and being present during runtime.
Image URL	Use this option to specify the URL for the image.
+ Add URL Parameters	Specify any parameters for the URL above.
Indicators	Use this multi-select box to configure indicators for the timeline event.

Option	Description
Additional Tooltip	Use this multi-select box to configure attributes for tooltip.

Example of Timeline Event Configuration



Grid

The Grid component displays a list of records in any configurable layout that best highlights different elements of the record. It has powerful capabilities for listing long textual values as well as images, indicator icons, buttons and star ratings.

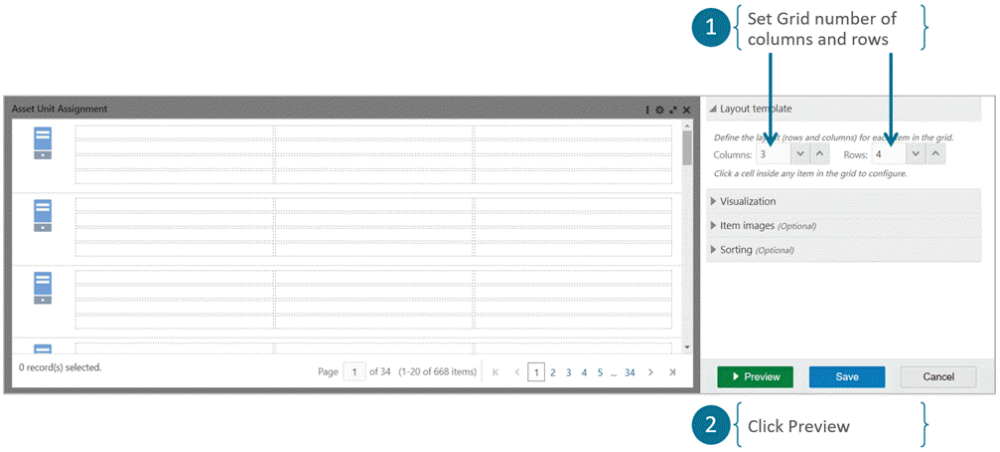
Specific Configuration

Grid Configuration Options

Option	Description
Layout Template	• Consists of multiple rows and columns where cells support any number of column/row spans. • Allows the display of multiple item types in the same cell, such as dimension and metric attribute values, input texts, custom buttons, star rating, and visual indicator icons. • To define the layout template, set the grid column and rows. • Click Preview to start configuring each cell in the layout template.
Add items in a cell	Click +Add item drop-down list and select one of the values: • Attribute values • Button • Text Input • Indicator Icon

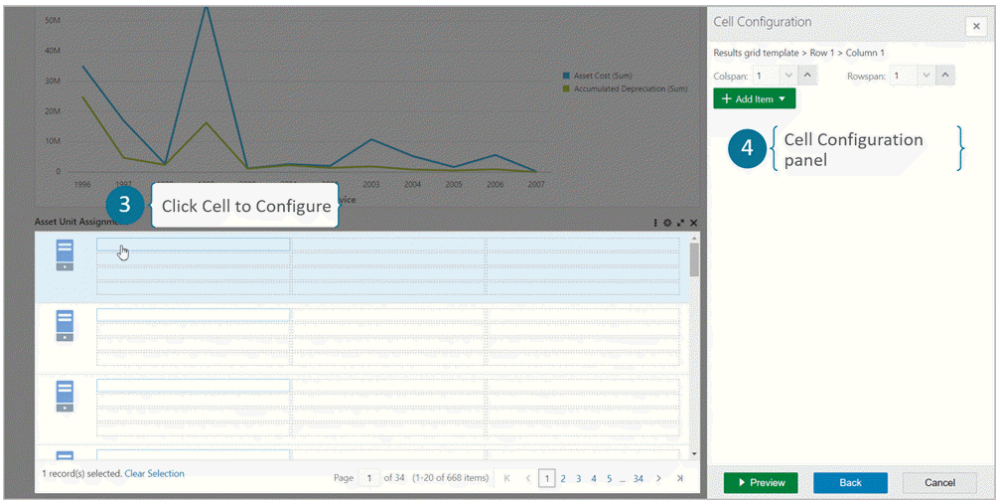
1. Set the number of columns and rows for the grid.

Setting the Number of Columns and Rows



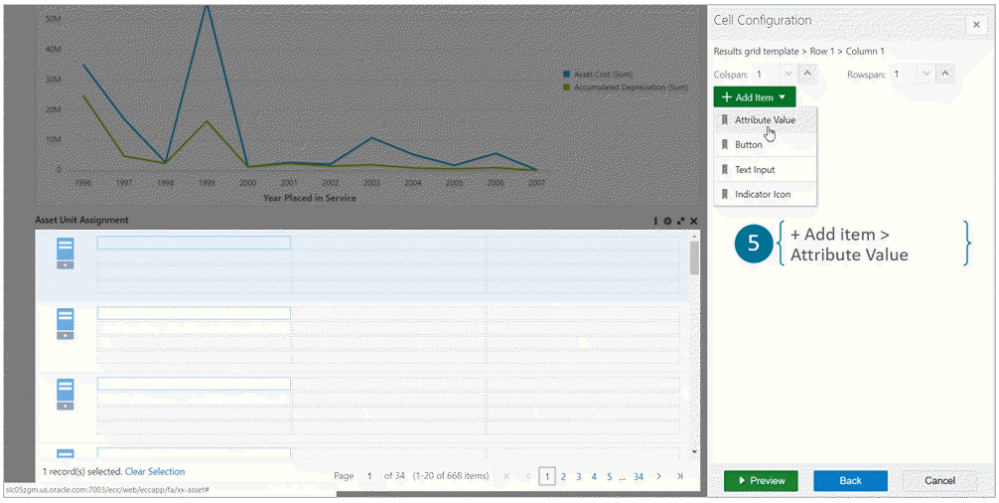
2. Click Preview.
3. Click a cell to configure it.
4. A Cell Configuration panel is shown.

Cell Configuration Panel



5. Click on "+ Add Item" and select 'Attribute Value'.

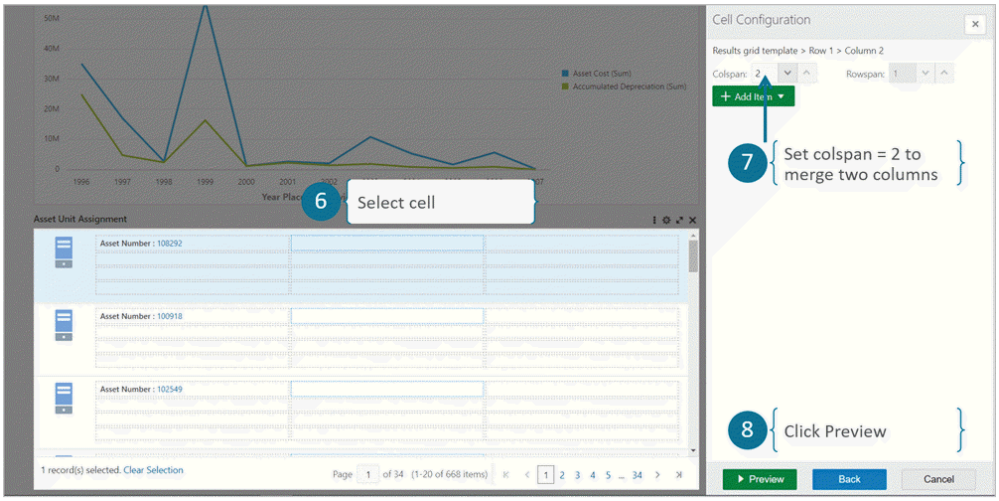
Selecting Attribute Value



6. Select a cell.

7. Set "Colspan" to 2 to merge two columns.

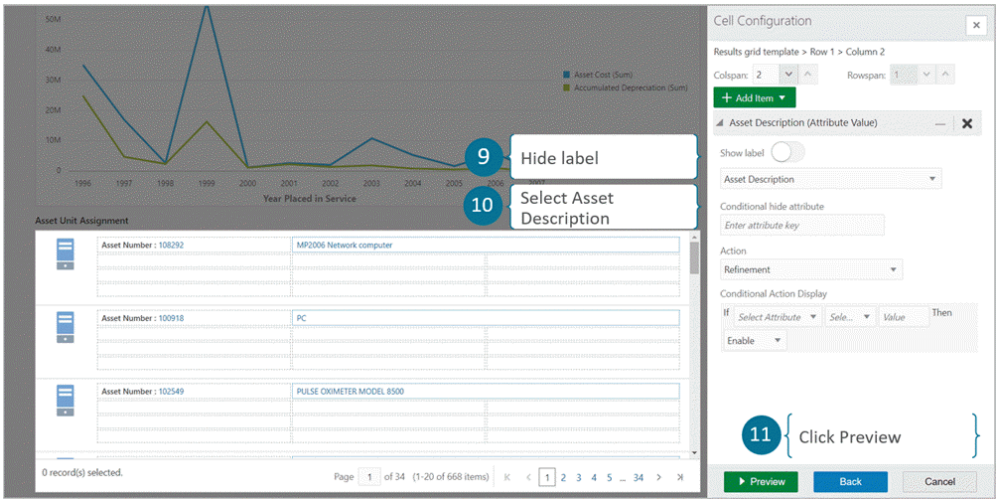
Merging Two Columns for a Cell



8. Click Preview.

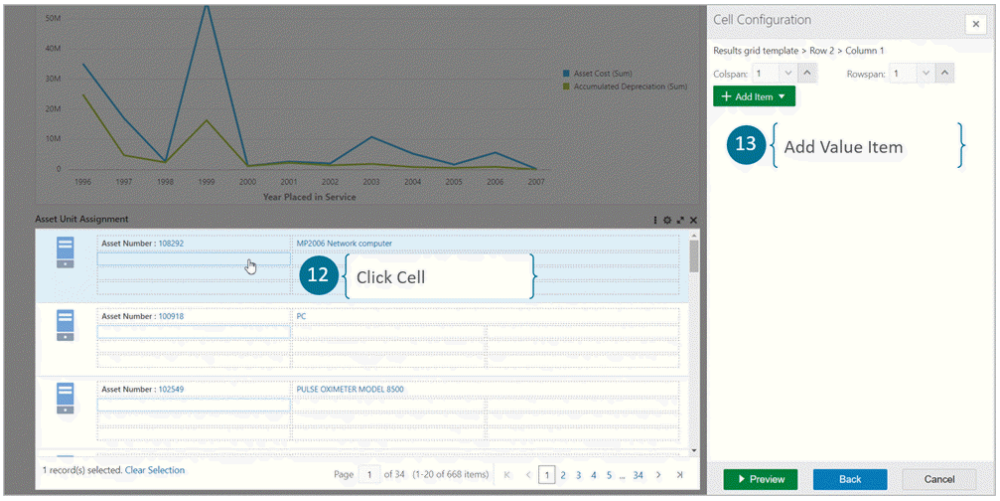
9. Toggle the "Show label" setting to hide the label.

Hiding a Label



10. Select Asset Description.
11. Click Preview.
12. Click a cell.
13. Select "+ Add Item" to add a Value item.

Adding a Value Item

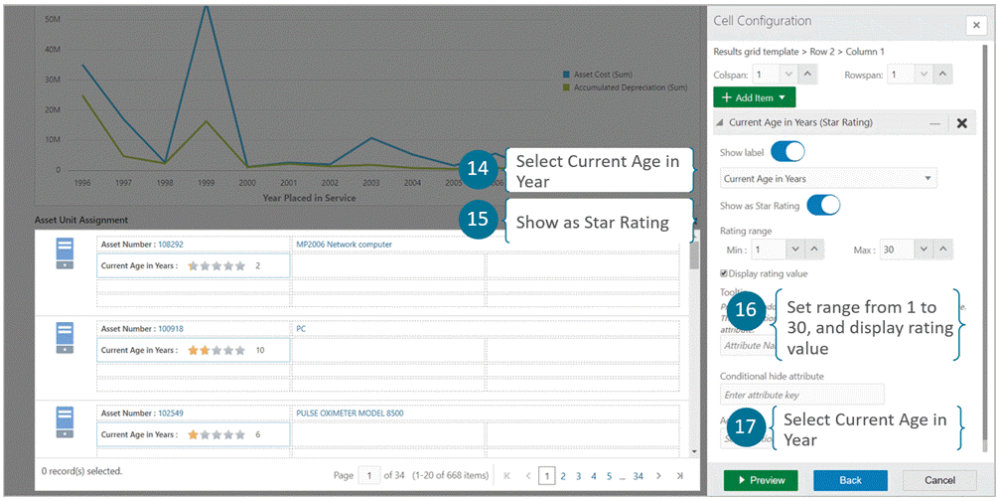


14. Select "Current Age in Years".
15. Select "Show as Star Rating".

16. Set the range from 1 to 30, and select "Display rating value".

17. Select "Current Age in Years".

Grid Star Rating Configuration

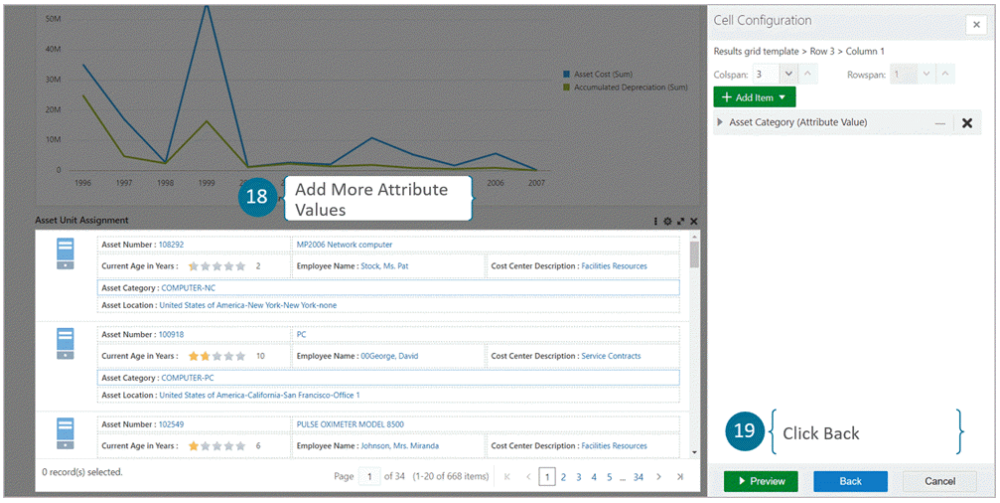


18. You can then add more attribute values.

Grid Visualization Option Attributes

Option	Description
Visualization	• To control component height, set the height in pixels. • Set the number of grid columns. • Set the number of items per page. • To enable Compare, check the "Enable Compare" checkbox. Add the Compare Header display name.

Adding More Attribute Values



19. Click "Back".

20. Click Visualization.

    Set the following visualization options:

    -

    - To control component height, set the height in pixels.

    - Set the number of grid columns.

    - Set the number of items per page.

    - To enable Compare, check the "Enable Compare" box.

      Add the Compare Header display name.

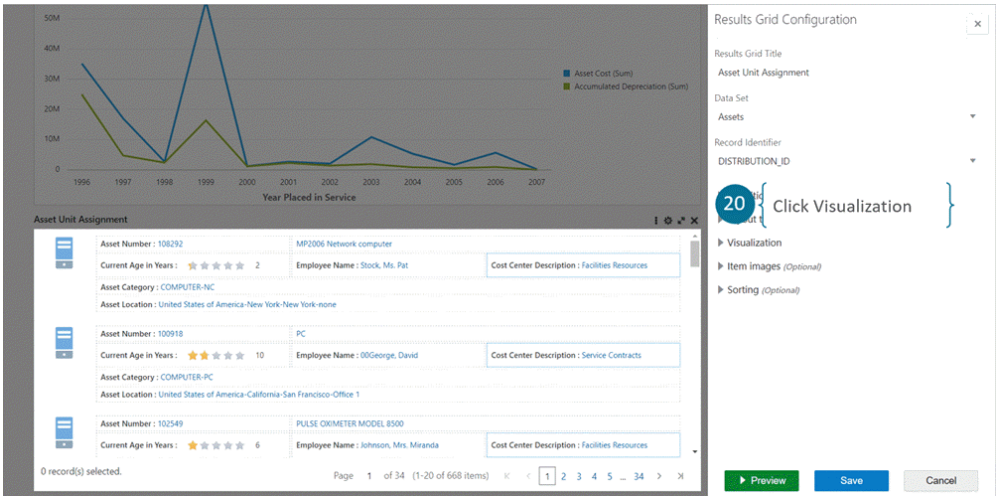
    - Set the record selection mode

      - Multiple: Multiple records of the grid can be selected at a time. This is the default option.

      - Single: Only one record can be selected at a time.

      - Disable: Record selection is disabled.

Selecting Visualization

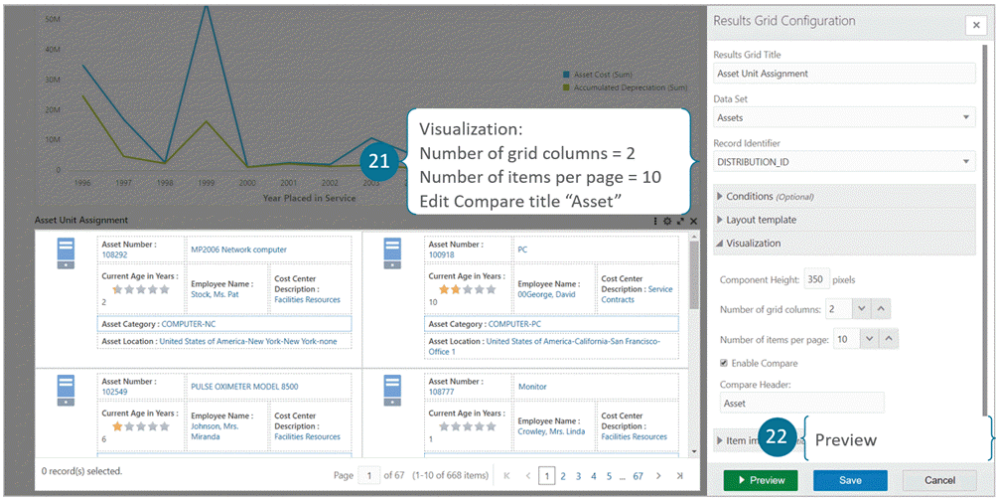


21. Visualization is set to the following:

    - Number of grid columns is 2
    - Number of items per page is 10

    Edit "Compare Header" 'Asset'.

Grid Visualization Configuration



22. Click Preview.

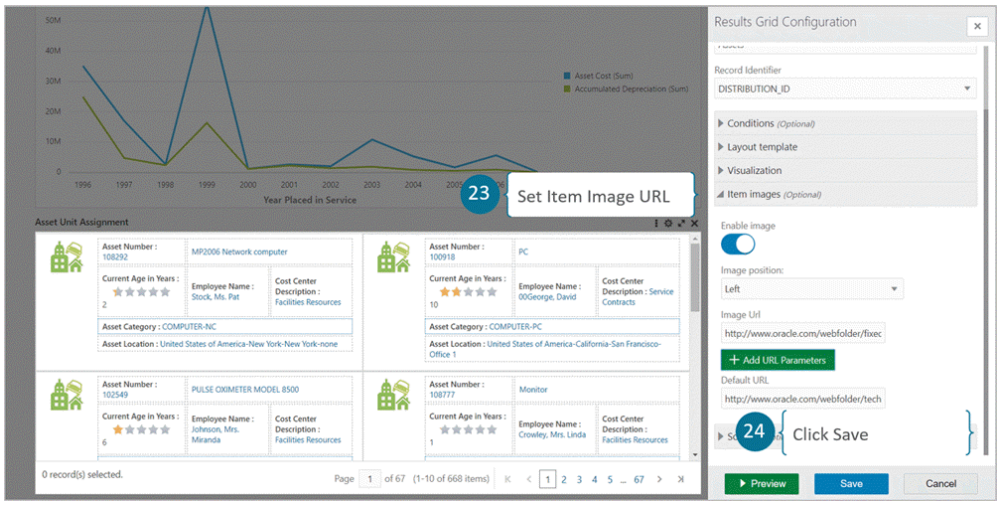
23. Set the Item Image URL.

Grid Visualization Configuration Options

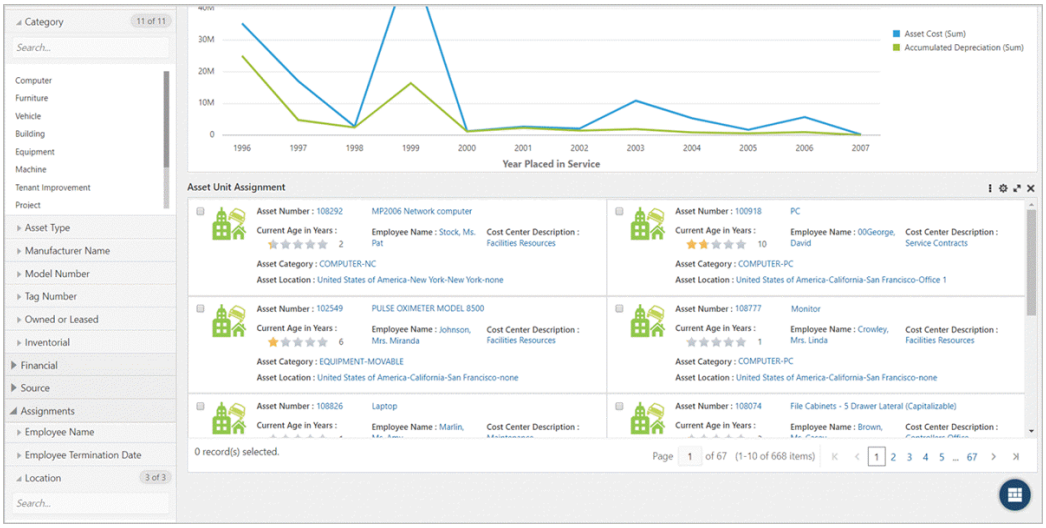
Option	Description
Item Image	Enable image, to allow displaying image in the grid • Set image position • Image URL

24. Click Save.

Grid Item Image Configuration



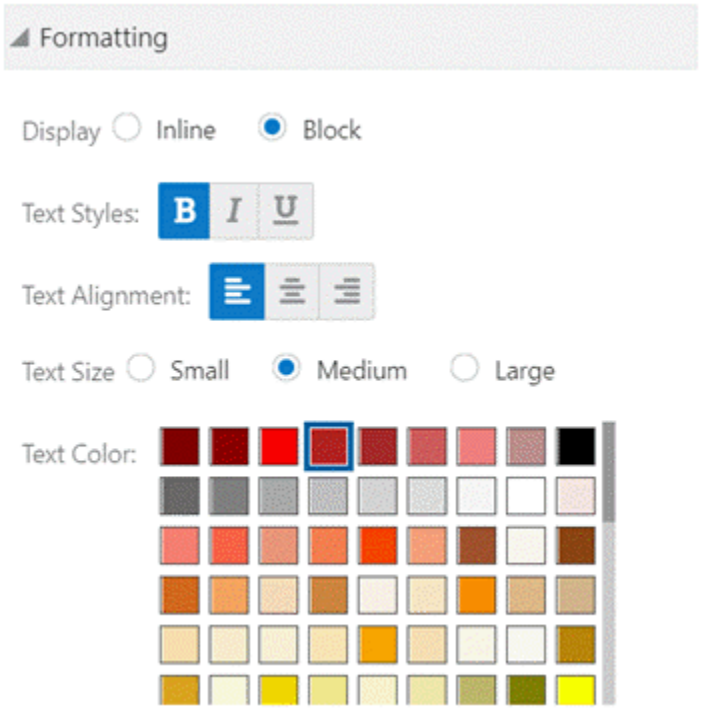
Grid Component



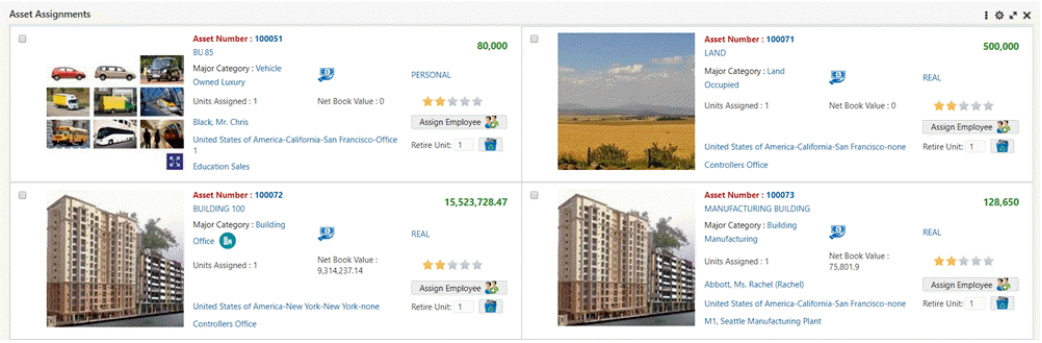
Grid Formatting Options

Option	Description
Formatting Note: Is part of attribute value configuration	Control the formatting of text in grid from 'formatting' accordion after selecting the cell Format the text with options: • Display > Inline/ Block: controls how the attribute value will be displayed in the cell in one line (inline) or different lines (block) • Text Style > Bold, Italic and Underline • Text Alignments > Left, Right, or Center • Text Size > Small (10), Medium (12), large (14) • Text Color > select from the color palette

Example of Text Formatting in a Grid



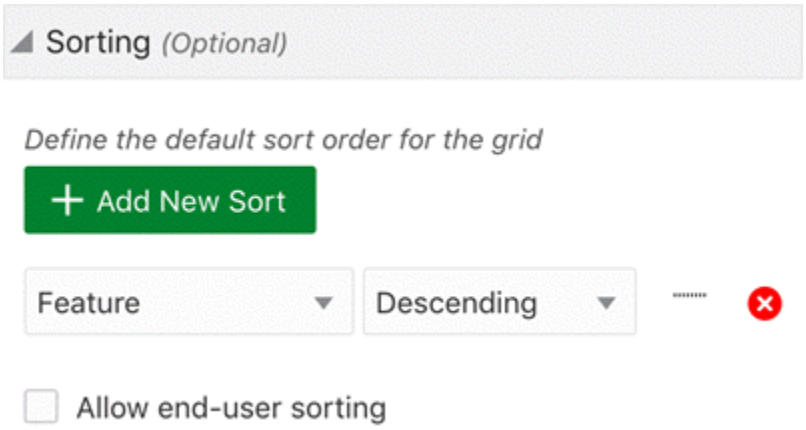
Text Formatting at Runtime



Grid Sorting Options

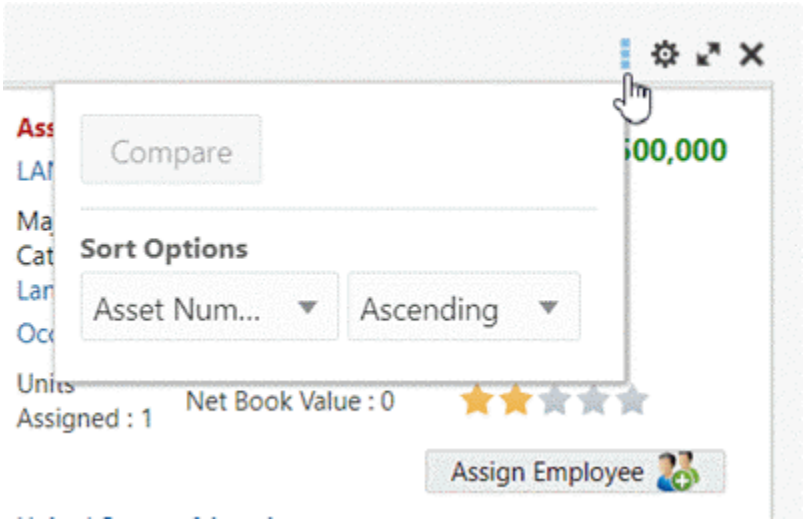
Option	Description
Sorting	• Allows user to define the default sort order to use for the grid • To define a new sort rule, click + Add New Sort • Select an attribute and Ascending or Descending sorting option • Allows end user sorting; enables the end user to change the default sorting at runtime

Grid Sort Option



A user can change the method of sorting using the Options icon.

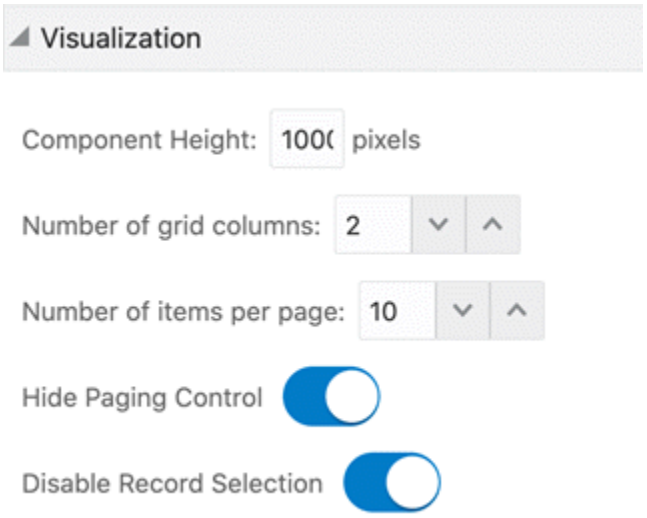
Allow End-User Sorting



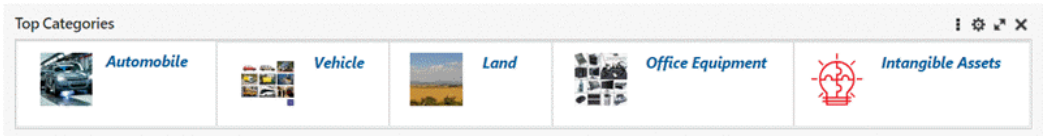
Grid Visualization Options

Option	Description
Visualization	• Hide Pagination Controls > Allow displaying the Grid without pagination bar • Disable Record Selection > Allow removing the record selection checkboxes at runtime

Example of Visualization Configuration

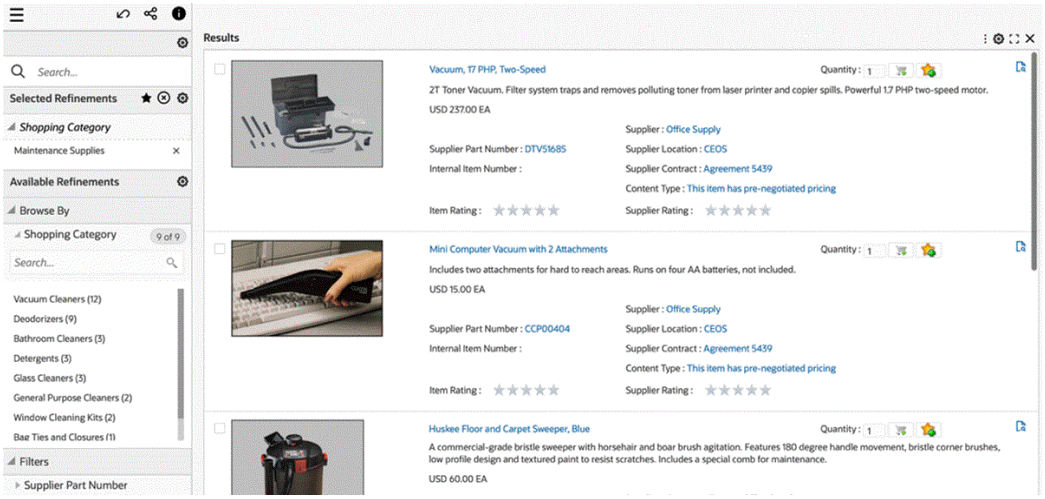


Example of Grid without Controls and Record Selection



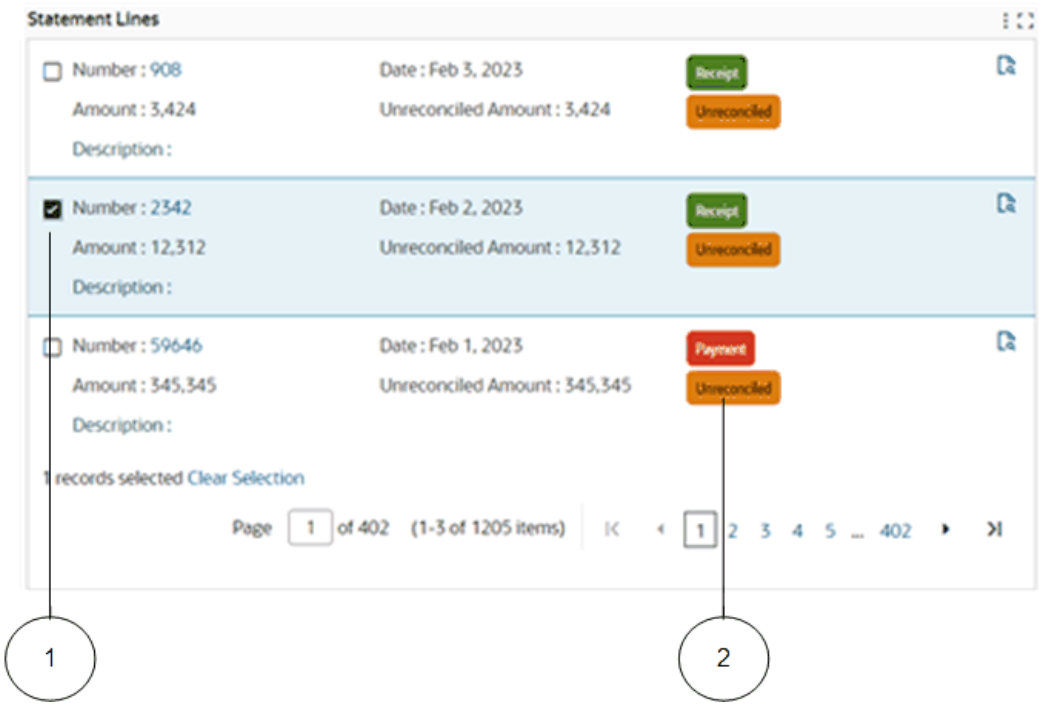
Beginning with V9, the Grid component is more compact. All attributes are top-aligned.

Example Grid with Compact Design



Beginning with V10, metadata-level color pinning is extended to the Grid component. Color pinning is used in the conditional formatting of categorical attributes in a grid.

Example of a Results Grid in Single Record Selection Mode with Conditional Formatting Enabled



1. Single record selected
2. Badge with a specifc color pinned to its attribute

In ECC V10, the Results Grid component's "Button" styling options have been enhanced. With the introduction of additional styling options of "Button" in the Results Grid component, designers are able to customize buttons to suit their requirements in ECC dashboard. These options provide more flexibility to the designer and help make the look and feel more intuitive to the end user.

The following are enhancements in V10:

- New Button "Size" option which allows designers to customise the size of the button. The default size is medium and is preselected.
- "Hide Runtime options" under the visualization accordion. This will hide runtime options, maximization, and the component border.

Beginning with V11, the Grid component shows all the selected records under a single view. The footer of the grid is enhanced to include a hyperlink when records are selected. Upon clicking the hyperlink, a user can access all the selected records under a single view. The selected records are overlaid on the original grid with a breadcrumb in the grid title to go back to the original grid.

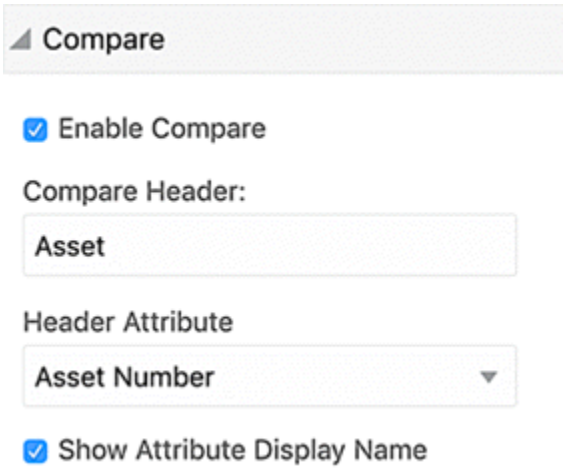
Compare Configuration

Configuration options for the Compare feature are listed in the table below.

Configuration Options for the Compare Feature

Options	Description
Enable Compare	Box to enable the Compare feature at runtime.
Compare Header	Allows adding static test as a Compare header.
Header Attribute	Displays the Compare header as a tokenized attribute.
Show Attribute Display Name	Flag to display both the attribute label and the value.

Compare Configuration



The options to display the column header are described in the following table.

Configuration Matrix

Header	Attribute Header	Display	Example
Empty	Empty	Record n	Record 1
Text	Empty	Text n	Item 1

Header	Attribute Header	Display	Example
Text	Attribute Name	Text: Attribute Value	Item: LCD Thin Panel Monitor
Empty	Attribute Name	Attribute Display Name: Attribute Value	Description: LCD Thin Panel Monitor

The options to control the display of attribute groups in the Compare window are listed in the following table.

Options for Display of Attribute Groups in the Compare Window

Option	Description
In Metadata Attribute Groups, Include in user actions	checkbox to enable groups to be displayed in Compare window
In Results Grid Configuration, Use Attribute Groups	checkbox to enable groups to be displayed in Compare window (Groups configured in metadata will be displayed)

The option to control the display of grid attributes in the Compare window header is listed in the following table.

Options for Display of Grid Attributes in the Compare Window Header

Option	Description
In Results Grid Cell Configuration, accessible in Compare and Record Details (Available for grid items: Text Input and Button)	checkbox to display grid item in Compare window header

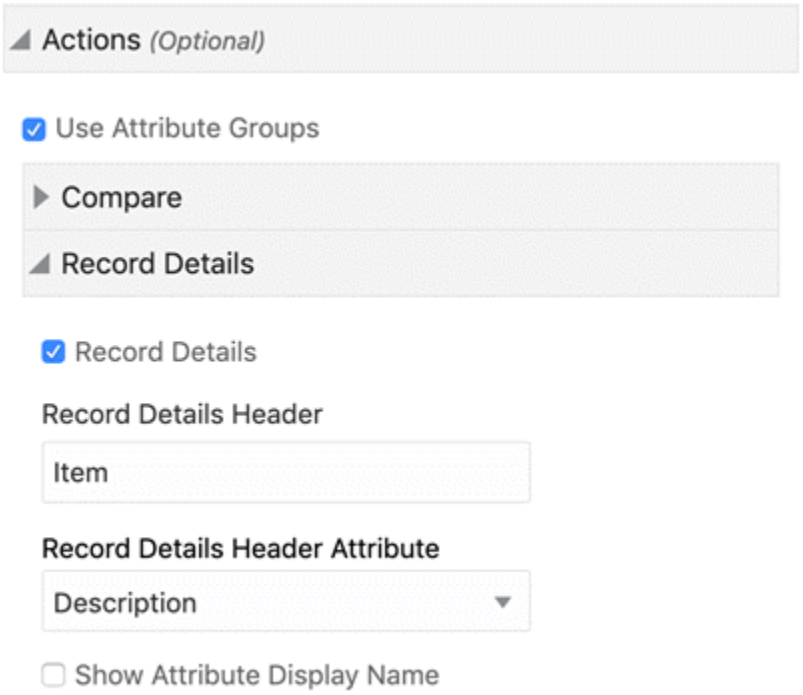
Record Details

The Record Details option allows you to view all the significant information of a record in a tabular format.

Configuration of Record Details

Options	Description
Enable Record Details	checkbox to enable record details
Record Details Header	Allow adding static test as a compare header
Header Attribute	Display record details title as a tokenized attribute
Show Attribute Display Name	Flag to display both attribute label and value

Record Details Configuration for Results Grid



The options to display the column header are described in the following table.

Configuration Matrix

Header	Attribute Header	Display	Example
Empty	Empty	Record n	Record 1
Text	Empty	Text n	Item 1
Text	Attribute Name	Text: Attribute Value	Item: LCD Thin Panel Monitor
Empty	Attribute Name	Attribute Display Name: Attribute Value	Description: LCD Thin Panel Monitor

The options to control the display of attribute groups are described in the following table.

Options to Control Display of Attribute Groups in Record Details

Option	Description
In Metadata Attribute Groups, Include in user actions	Box to enable groups to be displayed in record details
In Results Grid Configuration, Use Attribute Groups	Box to enable groups to be displayed in record details (Groups configured in metadata will be displayed)

The option to control the display of grid attributes in the record details title is described in the table below.

Option to Control Display of Grid Attributes in Record Details Title

Option	Description
In Results Grid Cell Configuration, Accessible in Compare and Record Details (Available for grid items: Text Input and Button)	Box to specify display of a grid item in the Compare window header.

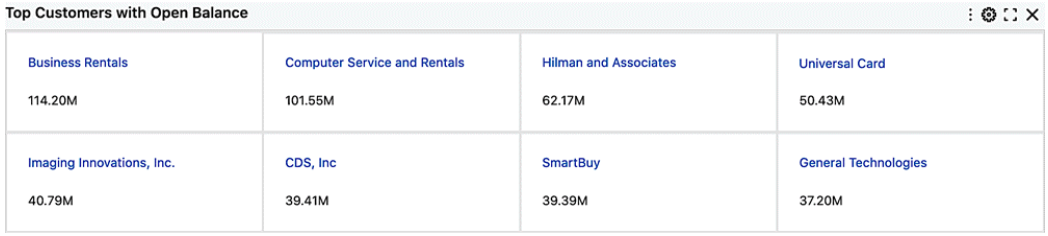
Aggregated Grid

The Grid feature supports the display of aggregated metrics. All dimensions in the grid control the aggregation level. Aggregation leverages all the visible grid items driven by attributes, including image and indicator icons. The Aggregated Grid feature has all the flexible layout and formatting options available for the grid.

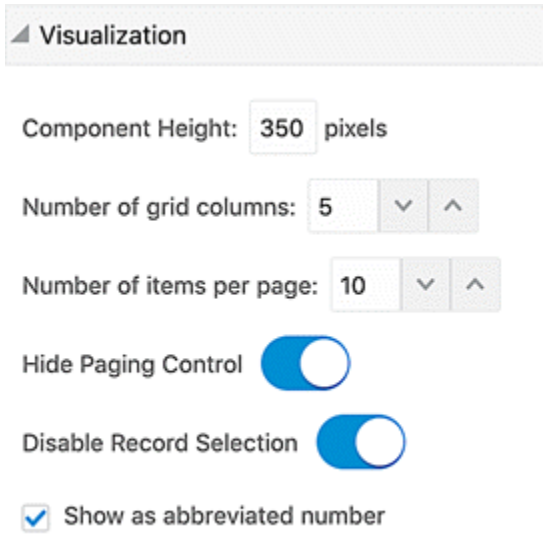
Users can not filter by aggregated value. Also, aggregated grids do not support the compare, record details, and attribute actions.

The Aggregated Grid feature supports language-sensitive abbreviation. Also, aggregate conditions and condition for display can be configured for Aggregated Grid.

Example of an Aggregated Grid



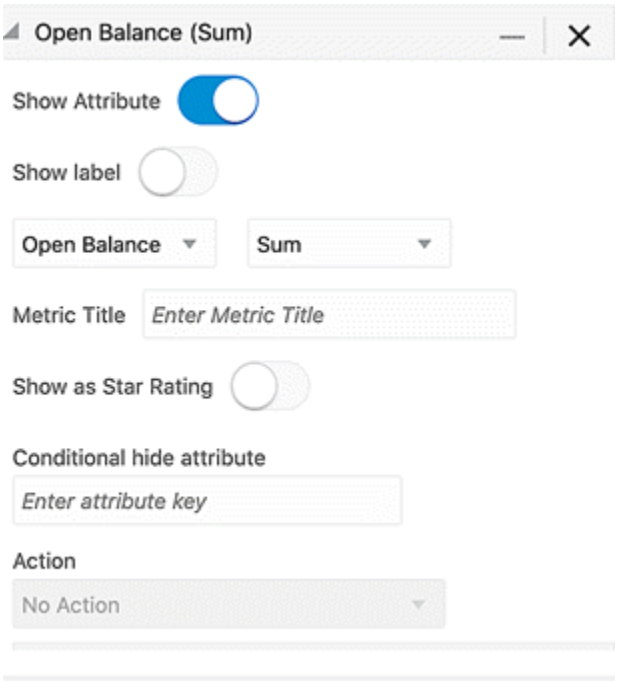
Visualization Options for Aggregated Grid



Aggregated Grid Configuration Options

Option	Description
Visualization	Show as Abbreviated Number Enable the flag to show abbreviated numbers instead of actual values for aggregated values. Actual values are displayed as a tooltip. The option is visible if at least one aggregated attribute is configured. You can change the default option by disabling the flag.
Add aggregated items in a cell	Click the +Add item list and select one of the values: • Attribute values • Select an attribute • Select an aggregation function

Configuration of an Aggregated Grid

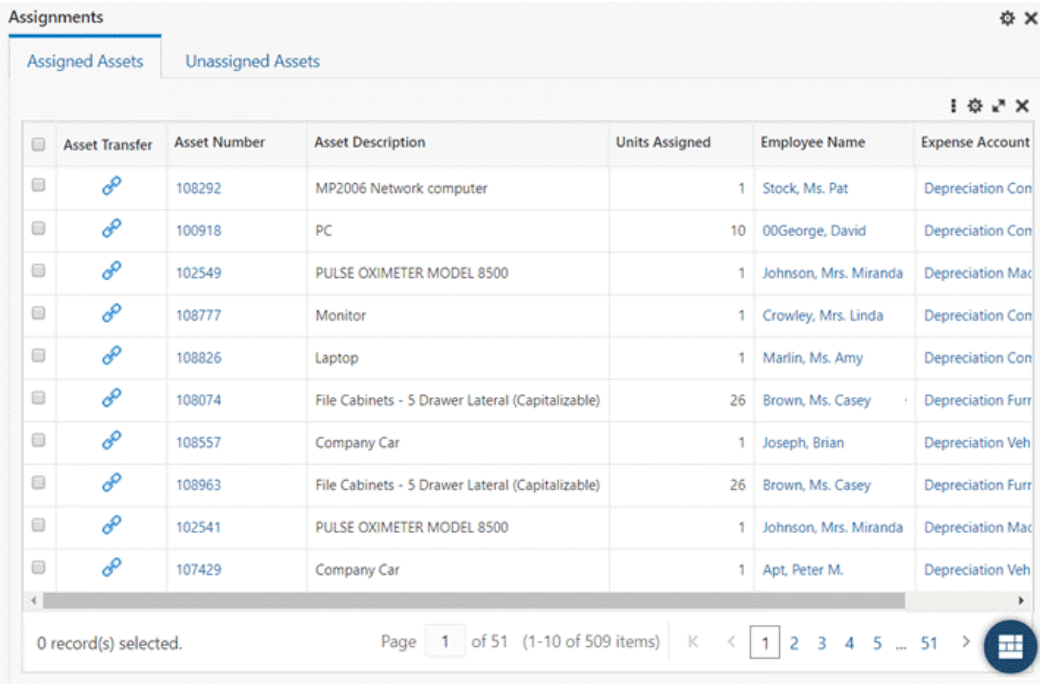


Layout Components

Tabbed Component Container

The tabbed component container allows you to group components on a dashboard. Containers cannot be nested within each other and you cannot move components inside or outside a tabbed component.

Tabbed Component Container



Configuration

The Tabbed Container groups components into multiple tabs. Each tab may contain one or more components.

By default, the container includes two tabs.

Configuration Options for the Tabbed Component Container

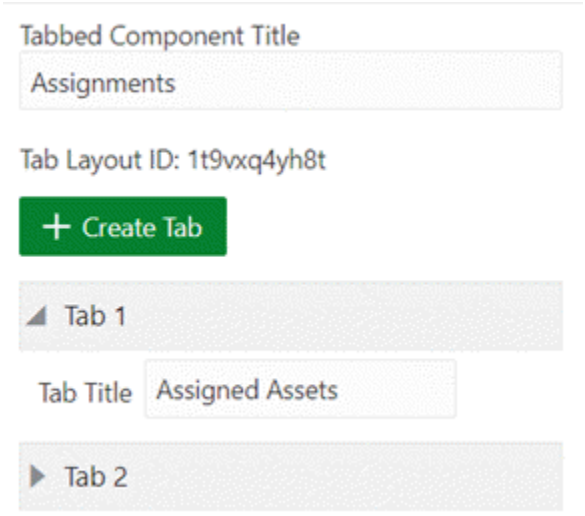
Option	Description
Title	Add a title for the component

Option	Description
+ Create Tab	To add a new tab
Tab Title	Add a title for the tab

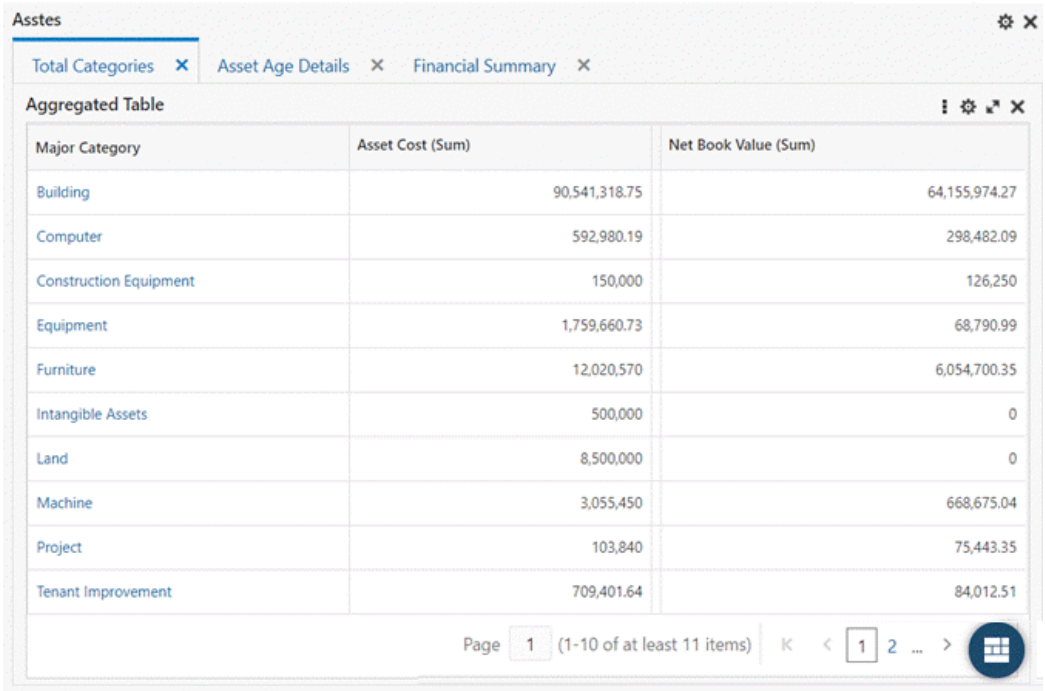
Only the administrator user at runtime can delete tabs and reorder the tabs within the Tabbed Container.

The Delete icon is displayed only if the tab contains more than two tabs.

Tabbed Component Configuration



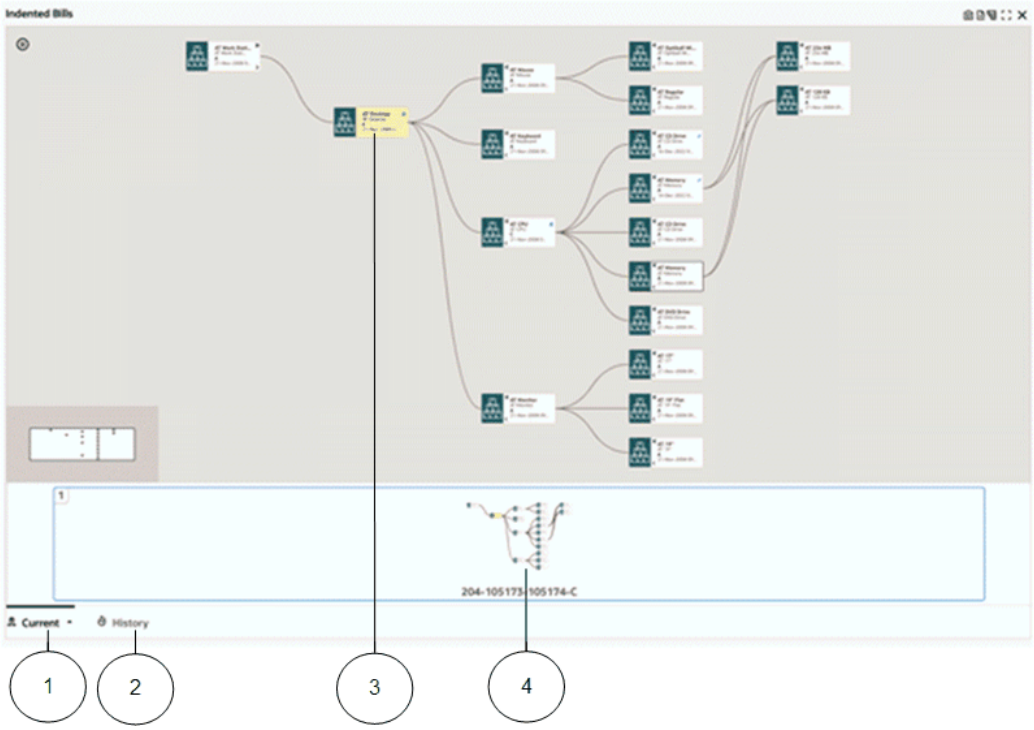
Deleting Tabs



Diagram

Introduced in V6, the diagram component provides immediate insights into a business flow or a process lifecycle through a network view. It allows a fast and efficient analysis and reporting of specific process flow. For example, tracking critical quality issues during the manufacturing and distribution cycle: a diagram simplifies tracing the origination of raw materials, manufacturing process, and shipping products to customers. Another example is tracking purchase order and related lots, work orders, and serial numbers for items, in which you can trace the relationships between one another.

Example of a Diagram



1. Current tab
2. History tab
3. Anchor node
4. Pagination control

You can zoom in to focus on an intermediate process or zoom out to get a perspective of the entire process.

Beginning with V9, the Diagram supports visualizing events from multiple data sets in a timeline view.

The diagram enables the designer/developer to configure the process flow/cycle flexibly; the main two elements in the diagram configuration are:

- Business entities, represented in the data sets
- Entity relationships, represented in the data set relationships

Configuration of diagram contains three major sections:

- Data set

- Data set Relationship
- Visualization

Data set configuration includes options to create business entities as data sets and represents these business entities as nodes. A designer can configure the attributes required for the diagram and can set text formatting and configure hyperlink actions for every attribute. These attributes are described in the table below.

Attributes for a Diagram

Category	Option	Description
Data Set	Add Data Set	Configuration for adding multiple data sets.
Data Set	Show on the node	Control to display node title.
Data Set	Node Title	Free text to set Node Title, defaults with data set name, if left empty.
Data Set	Node Identifier	Control to set granularity of data set. More than one attribute can be selected.
Data Set	Record Identifier	Attribute to be selected as record identifier.
Data Set	Conditions	Compound conditions with OR/AND operators among conditions/groups of condition.
Attribute	Show Attribute	Control to enable display of attribute display name on a node.
Attribute	Show Label	Control to enable display of attribute display name on a node.
Attribute	Aggregation	Used to define aggregation of an attribute.
Attribute	Apply Hierarchical Aggregation	Control to calculate hierarchical aggregation.

Category	Option	Description
Attribute	Custom Label	Used to define a business-friendly label for the aggregated metric.
Attribute Actions	Action	Control to define a hyperlink action.
Attribute Actions	Display Action Description in Tooltip	Control to display tooltip for Action (Available when Action is "Hyperlink").
Attribute Actions	Open link in new window	Control to open actions in new browser window (Available when Action is "Hyperlink")
Attribute Actions	Display in a Pop-Up	Control to open actions in ECC Pop-up (Available when Action is "Hyperlink").
Attribute Actions	Conditional Action Display	Control for defining when an Action should be available.
Indicators	Show as item image	Control to show indicator as an image for the node.
Indicators	Image URL	Control to provide URL in order to retrieve a thumbnail image. URL can include attribute values as parameters.
Indicators	Default Image	Control to display default image when another image is not available.
Indicators	+ Add URL Parameters	Control to add parameter for the URL above.
Indicators	Tooltip	Control to provide additional detail for the attribute value.
Indicators	Action	Control to define a hyperlink action.
Indicators	Display Action Description in Tooltip	Control to display tooltip for Action (Available when Action is "Hyperlink").
Indicators	Open link in new window	Control to open actions in new browser window (Available when Action is "Hyperlink").

Category	Option	Description
Indicators	Display in a Pop-Up	Control to open actions in an ECC Pop-Up (Available when Action is "Hyperlink").
Indicators	Display in a Drawer	Control to open actions in ECC Drawer (Available when Action is "Hyperlink").
Indicators	Display inline	Control to open actions in as an overlay on parent component (Available when Action is "Hyperlink").
Indicators	Conditional Action Display	Control for defining when an Action should be available.
Formatting	Text Style	Control to set font style of attribute label, value displayed on node.
Formatting	Text Size	Control to set font size of attribute label, value displayed on node.
Formatting	Text Color	Control to set font color of attribute label, value displayed on node.
Record Details	Record Details Header	Text to display as record details pop-up header.
Record Details	Record Details Header Attribute	Attribute to be displayed as record details pop-up header.

Data Set Relationship Configuration

The Diagram component requires configuration of relationships among data sets to display appropriate links among the nodes.

Configuration allows defining multiple relationships among data sets based on one or more attributes. It also allows defining relationship between same data sets but with different attributes

A diagram displays related nodes and links in a single page and automatically creates another page to show other related nodes.

Data Set Relationship Configuration Options for the Diagram Component

Option	Description
Select Source Data Set	Control for selecting one of the configured data sets as source data set of a relationship.
Select Target Data Set	Control for selecting one of the configured data sets as target data set of a relationship.
Select Attribute	Control to select attributes for defining relationship between the data.

Beginning with V7, the concept of an alias provides the ability to define a data set more than once in a diagram. Every alias is treated as a virtual data set and can be configured as a different node.

> **Note:** The alias name for a data set node in a diagram is used as a key for diagram processing and cannot be translated. Consequently, when a user hovers over a data set node icon in the user interface, the alias name is always displayed in Roman letters.

Hierarchical View

An Hierarchical View accordion allows a user to use a hierarchical query instead of a diagram and subsequent hierarchical query settings, including the ability to create a data set view.

Hierarchical View Options for the Diagram Component

Option	Description
Use Hierarchical Query	Allow the diagram to represent a hierarchical structure in a vertical layout. Tree/vertical layout is supported with expanding and collapsing the hierarchy level by level for better readability.
Show level	Control to show the level.

Option	Description
Display as card type	Control to align nodes as a card; that is, center-align the nodes.
Create Data Set View	Control to create Data Set View from the Hierarchical Query.

Visualization Configuration

Visualization configuration allows designers to control the UI clutter while improving the performance. Designers can set an appropriate number of nodes to be displayed and control the maximum number of pages a diagram can be spanned across.

Beginning with V8, the Diagram component supports a "Row Expander" feature, which allows business users to switch between a diagram display and a multilevel hierarchy layout display.

Beginning with V9, the Diagram component supports three kinds of visualization:

- Diagram: To show insights into a business flow through a network view
- Timeline: To show the chronological sequence of related events in a business flow
- Hybrid: To show both - the business flow through a network view and chronological sequence of related events in a timeline view

Configuration options are described in the table below.

Visualization Configuration Options for the Diagram Component

Option	Description
Visualization Option	List for selecting the type of visualization for the component.
Component Height	Number to set the height of component in pixels.
Page Limit	Upper limit on number of pages allowed to be displayed.

Option	Description
Nodes Limit Per Page	Upper limit on number of nodes displayed on page (Appropriate number has to be configured in order to avoid UI clutter).
Timeline as Default View	Control to set Timeline view as the default view in Hybrid Case (Applicable only when the component's visualization type is Hybrid).
Visualize Node Grouping	Visualizes grouped nodes that control the display aspects of the diagram nodes at a different level of detail; also allows for the grouping redundant nodes.
Show Zoom Window	Enables the display of the overview window rendered within the diagram.
Row Expander as Default View	Control to set 'multilevel hierarchy layout' as the default display option (Applicable only when the component's visualization is either of type Diagram or Hybrid).
Default Display Option	Control to select the default dataset for the row expander context.

Configuration options specifically for the Timeline View are listed in the table below.

Visualization Configuration Options for the Timeline

Option	Description
Timeline Overview	Enable the draggable timeline and overview to pan around the Viewport.
+ Add Event	Click on the **Event** button to add timeline events.
Event Name	Enter the event title that will be displayed on the timeline.
Title	Select the main attribute that will be timeline event.

Option	Description
Description 1	Specify a description attribute.
Description 2	Specify another description attribute.
Start Date	Event start date.
End Date (optional)	Event end date. If this option is configured, the timeline displays the event duration based on the start and end dates.
Apply Event Identifier	Enable this option to show only unique events based on Title, Description 1, Description 2, Start Date, and End Date
Show Label	Box to show or hide the attribute label.
Display Image	Enable this option for displaying an image in the timeline. Note that the image will be visible only in the case of multiple events being configured and are present during runtime.
Image URL	The URL for the image.
+ Add URL Parameters	Any parameters for the URL above.
Indicators	Multi-select box to configure indicators for the timeline event.
Additional Tooltip	Multi-select box to configure attributes for tooltip.

In the case of the Hybrid visualization, a designer is presented with separate accordions for the timeline view and diagram view, with an option to set default view in the common accordion.

Row Expander Freeze

The Row Expander Freeze feature, introduced in V10, allows the designer to freeze or lock attributes so that when a user scrolls the context of the node is maintained. This feature is available for both the network diagram and hierarchical queries. Every attribute in the configuration for diagram has a flag for "Freeze in row expander" so a

designer can set any attribute to be frozen/locked in row expander by selecting this box. Once an attribute is checked as "Freeze in row expander," that attribute will be frozen, When an end user scrolls within a row expander view, they will still see the frozen attributes. A light grey background appears for the frozen attributes. If the designer selects an attribute which is in the middle or last and chooses to freeze it, then that attribute will move to the front.

Search Within

Beginning with V10, the Diagram component includes the Search Within capability. This feature was previously only available for Hierarchical Query. This feature allows users to investigate a subset of the larger network diagram they are viewing to focus on a problem area, perform root cause analysis, or identify an opportunity within the overall network diagram. Users can view a specific business flow and apply a filter to create a diagram to visualize that flow. This filter is known as the "Driving Filter." After applying a "Driving Filter," the user can use a "Search Within" filter to refine the diagram further and focus on a subset of the diagram. Nodes that satisfy the "Search Within" condition are highlighted with a more prominent and noticeable border color, and the path which connects (directly/indirectly) is also highlighted. The nodes and links that are not connected to qualifying nodes are dimmed. This makes the nodes and the path stand out, and the user can then focus on this subset of the diagram while being in the same larger context.

The following is the result once a Search Within filter has been applied:

- The nodes which have satisfied the criteria are highlighted with a dark blue border.
- The nodes which do not directly satisfy but have child/ children nodes that do are left as is.
- The nodes that don't satisfy either of the above are dimmed out.
- The nodes with multiple paths connected to them are highlighted in blue.
- Users can click on the funnel icon and view what Search Within filters have been applied.

Highlight Path versus Search Within in a Network Diagram

The Highlight Path feature is used for a particular node in the network diagram. Users can see the chosen node and anything connected to it only; anything that is not directly or indirectly connected to this node is dimmed. The highlighted path is applied only to one node. The Search Within feature searches for any attribution on any node within the network diagram. Therefore, more than one node can satisfy the criteria, and all the nodes directly or indirectly connected to these nodes (that satisfy the criteria) are highlighted. Also, the Traversal filter, in the Available Refinements component, behaves like Search Within from Network Diagram, with the only difference being that the funnel icon is not displayed. Instead, the filter appears in the Selected Refinements box.

Search Within in a Hierarchical Query versus Search Within in a Network Diagram

With Hierarchical Query, because of the presence of only one dataset and therefore all nodes have the same attributes, whatever node satisfies the criteria will be displayed along with nodes that do not directly satisfy the criteria but have a child or children that do (although these nodes are dimmed). Any node that neither directly satisfies the filter criteria nor has a child or children that do will be removed from the display.

In a Network Diagram, because of the possibility of the presence of multiple datasets and therefore different sets of attributes, only the node that satisfies the filter criteria will be highlighted. Even in cases where the Network Diagram has only one data set the node that satisfies the filter criteria will be highlighted.

What happens when the user applies a Search Within from the network diagram and then, from available refinements, applies another Traversal filter or another filter from the same dataset?

Search within is a local component. Whenever the selected refinement is affected by the user either by applying another traversal filter or other filter from the same dataset, or by removing a filter, the Search Within filter is removed. In addition, the item filter selected from available refinements is then applied. Thus, if a Traversal Filter is applied from available refinements, then that filter replaces the prior Search Within filter.

What happens when a user applies a traversal filter from available refinements and then from the network diagram applies a Search Within filter?

The Search Within filter will be applied on top of the Traversal Filter. The two filters will form an AND condition if the Search Within fitler was applied on a different attribute, and will form an OR condition if the Search Within was applied on the same attribute.

How can a user reset a Search Within filter?

A user can click on the funnel icon to view the Search Within filters as in a Hierarchical Query. Also as with a Hierarchical Query, a user can click on the Clear button to reset the diagram to its initial state and remove the Search Within filter. This reset affects only the Diagram and not the whole page as the Search Within filter is local and only affects the Diagram/ Hierarchical Query component.

Traversal Filter in Hierarchical Query

Beginning with V9, the concept of a "Search Within" filter, that is, a local filter to the Diagram within the Hierarchical Query, is supported. This filter is local to the Diagram component and will not affect other components.

Once a "Search within" filter has been applied on a node, then the root node and the nodes that satisfy the condition are displayed. Any nodes that do not themselves satisfy the condition but have child/children nodes that do are also displayed but grayed out.

A funnel icon is displayed to indicate that a "Search Within" filter has been applied.

The following is an example of how a user can create a traversal filter in an hierarchical

query:

1. Right-click on any node and select **Show Details**.

2. Select any attribute value and then choose **Search Within**. This creates a local filter and that is only applied to current diagram locally. For example, select 'Manufacturing' as the Organization and click **Search Within**. You can select multiple attribute values as well.

3. Upon the selection of the "Search Within" filter, only the nodes that satisfy the condition, that is, have 'Manufacturing' as the Organization, are displayed. Those that do not have "Manufacturing as the Organization but have child/children nodes that do are displayed but grayed out.

The funnel icon denotes that the "Search Within" filter is applied.

You can click on the funnel icon to view the details of the "Search within" filter. You can click on the funnel icon and then the **Clear** button to remove the "Search Within" filter.

If you selects a new node as the anchor node, then the "Search Within" filter is removed as well.

Export to PDF

Users can export a hierarchical query diagram to a PDF.

Users can see that the "Search Within" is applied and, in the example above, see that the node with Organization that is not 'Manufacturing' has a different background to differentiate it.

Row Expander

The funnel icon indicates that a "Search Within" filter has been applied. A node with an Organization that is not 'Manufacturing' has a different background to differentiate it. When a user clicks on the funnel icon, the applied filters are shown, as in a Network diagram.

Users can remove the filter by hovering over the filter and clicking the **Clear** button in the Applied Filters window.

Considerations for Hierarchical Query

When a new node is selected as the anchor node, then the "Search Within" filter or filters are removed.

The "Search Within" filter or filters are not part of the Selected Refinement component and likewise do not impact or refine other components.

A user must filter only on the attribute/value found in a given node. A user cannot edit the filter as well in V9. User capabilities will be extended in a future release.

Aggregations are impacted when the user filters out nodes.

A user can apply multiple filters only in one attempt. That is, if a user wants to apply a

filter of location 'US' and total reports '4', then the user must find a node that has both values and apply the two filters together.

If a user applies one filter and then applies another filter, then the most recent filter will override the previously-applied filter. Both this point and the preceding one will be addressed in a future release where users will be allowed to choose from facets instead of finding the node with the required value.

In the row expander view, the display option list is not displayed for the hierarchical query diagram view.

Filters displayed at the top of the row expander in a hierarchical query diagram view are wrapped.

In V10, the Hierarchical Query design is changed. Updates include a new icon to indicate the Hierarchy. When this Hierarchy icon is clicked, the respective node becomes the anchor node. A deep link traversal filter is also introduced.

Starting with V11, designers can configure a Hierarchical Query to display a default image if there were no other image to be found for a node.

Support for Hierarchical Aggregation in Hierarchical Query

Beginning with V11, Hierarchical Query supports Hierarchical Aggregation to present a consolidated view of parent- and child-level information under a single tree structure. The row expander view also supports hierarchical aggregation.

Note the following:

- Node-level aggregation applies to all nodes, and it signifies the consolidation specific to an individual node.
- Hierarchy-level aggregation applies to parent nodes, and it signifies the consolidation encompassing both the node and its child nodes.
- The introduction of custom labels helps make a clear distinction between node-level aggregation and hierarchy-level aggregation.
- Custom labels can only be defined for aggregated attributes.
- Hierarchical aggregation is calculated when the following conditions are met:
    - Node identifier is defined.
    - Apply Hierarchical aggregation" is marked.
    - The Hierarchical Query view is selected.

Deep Link Traversal Filter in Hierarchical Query

Until ECC V9, Traversal Filter via Deep Link was possible only in a network diagram

and not in a hierarchical query. This V10 feature brings the same functionality in a hierarchical query so that designers can set up the dashboard in way which leads business users to the intended diagram with all the relevant driving and traversal filters without the users having to apply those themselves. Designers can submit a deep link in a hierarchical query just like in a network diagram; that is, a designer can append the "istraversal" flag with the value of 'true in the deep link URL for the respective attribute.

For example:

```
eccDeepLink=[{"ECC-BOM-
ASSCOMP":[{"attributeKey":"ORG_CODE","values":["V1"],"operator":" ="},
{"at
tributeKey":"COMPONENT_NAME","values":["AT Work
Station"],"operator":"=="},{"attributeKey":"ALTERNATE_BOM_DESIGNATOR","
val
ues":["Primary"],"operator":"=="},{"attributeKey":"REVISION","values":
[1],
"operator":"==","isTraversalFilter":true}]}]
```

ECC Pop-up

Beginning with Oracle Enterprise Command Center Framework V6, the ECC Pop-up component allows users to perform an Oracle E-Business Suite transaction on an Oracle Application Framework page within the dashboard itself. Users can review additional information or access a third-party website related to any record in the pop-up without navigating to any other page.

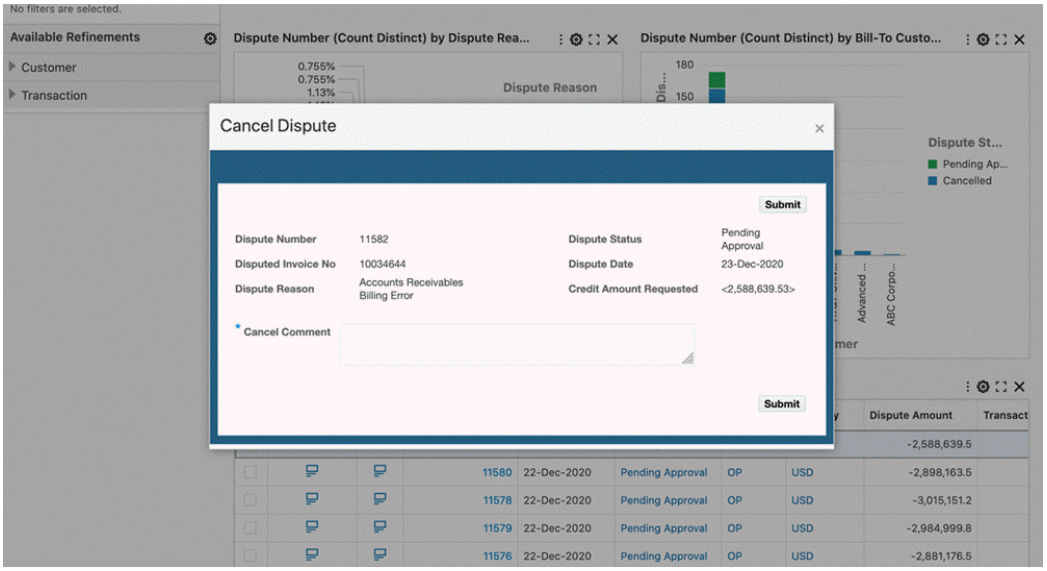
> **Note:** Some external websites may not be loaded because of browser security issues (such as cross-origin requests or mixed content issues).

An ECC Pop-up can be configured for hyperlink actions in the Diagram, Results Table, and Grid components to support display of the hyperlink contents in a pop-up.

Designers can combine the push model feature with ECC Pop-up so that the dashboard can immediately reflect the changes applied by a user's action in the ECC Pop-up.

> **Note:** Adding the property `eccPopupMode=Y` to the link in the respective component's configuration will hide the Oracle Application Framework global headers, footers, and buttons.

Example of an ECC Pop-up



Configuration Options for the ECC Pop-up

Option	Description
Display in a Pop-up	Available only for Action Type: Hyperlink.
Pop-up Title	Free text for setting a header/title on ECC Pop-up. Ex: 'Cancel Dispute'.
Lock Dashboard	This option locks the dashboard controls when ECC Pop-up is opened.
Refresh on Close	This option allows the configuration of refreshing dashboard contents or the entire dashboard.
Refresh Contents	Refreshes contents of every component in the dashboard that allows the changes made in pop-up to be reflected.
Refresh Dashboard	Refreshes the page to reflect changes.
Action Url	Hyperlink for the pop-up contents.

Option	Description
URL Parameters	Allows configuration of attribute values as parameters to the hyperlink.

ECC Drawer

In ECC V8, the Results Table, Grid, and Diagram components support linking contents through an ECC Drawer. A drawer is a panel that slides in and out from the side. It helps users complete tasks faster and more efficiently. As an example, users can use drawers to navigate to another ECC page or any other Oracle Application Framework page. Using an ECC Drawer is recommended linking another ECC dashboard.

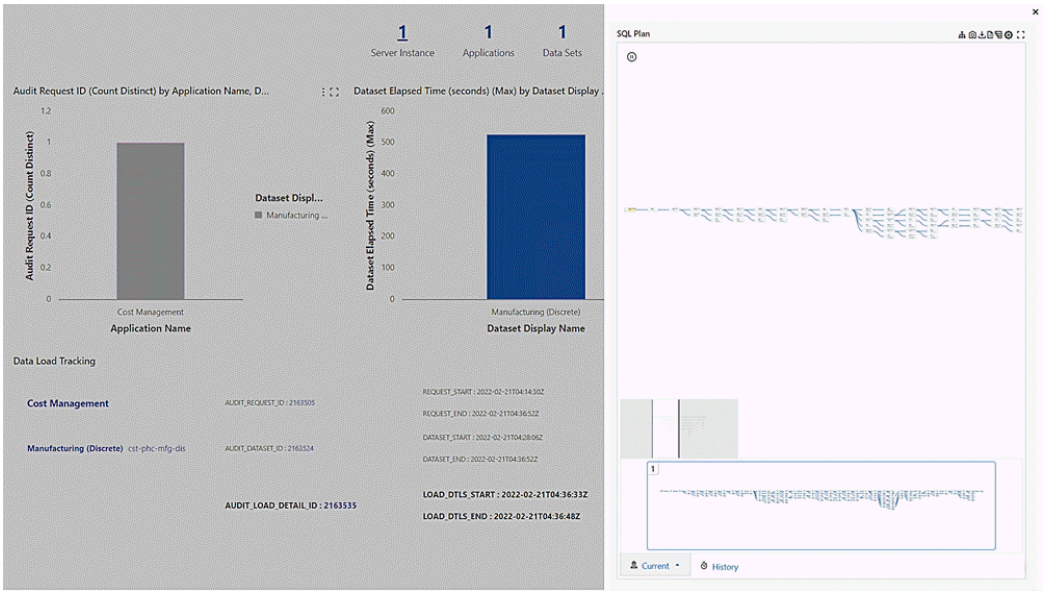
ECC Drawer is accessible from any hyperlink configured on Results Table, Grid, and Diagram.

Designers can combine the push model feature with ECC Drawer so that the dashboard can immediately reflect the changes applied by the user's action in the ECC Drawer.

Beginning with V11, the default size of the drawer component is set to 90% of the page size. An end user has the ability to reduce the size to 50%.

The configuration of an ECC Drawer is the same as that for an ECC Pop-up. Note that by default, the dashboard is locked for the drawer, and the option in the configuration is disabled.

Example of an ECC Drawer



ECC Inline View

Beginning with V9, the Results Table and Diagram components support linking information through the Inline View component. An inline view acts an overlay on top of the parent component. It helps the user to access related information without leaving the original context, leading to shorter turnaround time and better user experience. Through an intuitive breadcrumbs feature, an inline view provides seamless navigation between the parent, or source component and the target component. The primary use case for an inline view is providing access to context-specific information from detailed components: Results Table, Results Table-based Timeline (Single Dataset), Diagram-based timeline (Multiple Datasets).

Supported Targets for Source Components

Source Component	Supported Detail Target Component
Results Table	Results Table, Single Dataset Timeline, Multi-Dataset Timeline
Diagram	Results Table, Single Dataset Timeline, Multi-Dataset Timeline
Single Dataset Timeline	Results Table, Single Dataset Timeline, Multi-Dataset Timeline

Source Component	Supported Detail Target Component
Multiple Dataset Timeline	Results Table, Single Dataset Timeline, Multi-Dataset Timeline

A Single Dataset Timeline is a timeline that shows events from a single dataset. Such timelines are based on the Results Table component.

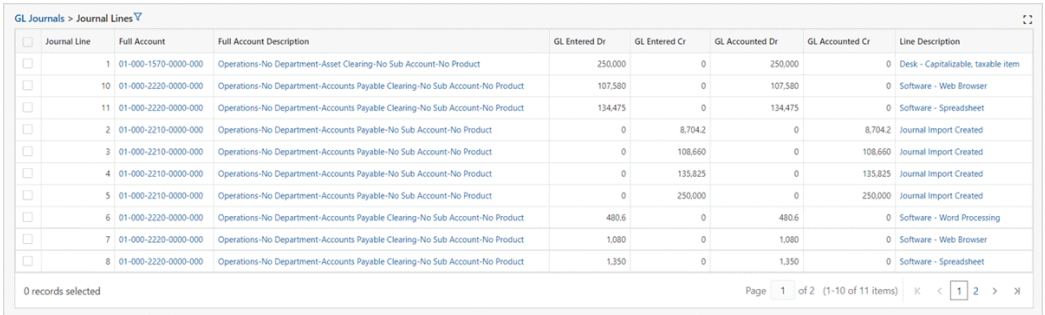
A Multiple Dataset Timeline is a timeline that shows events from multiple datasets. Such timelines are based on the Diagram component.

The inline view is accessible from any attribute-level hyperlink action configured on a Results Table or Diagram.

For example, an inline view could have a source or parent component of General Ledger Journals in a results table. The target component could then be the Journal Lines, also as a results table.

Example of an Inline View: General Ledger Journals Source Component

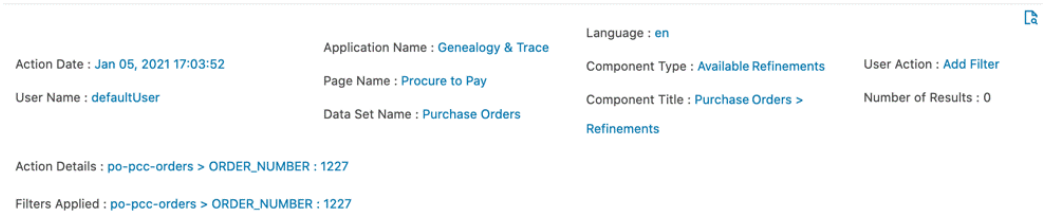
Example of an Inline View: Journal Lines Target Component



Support for Time Stamps and Time Zones

Business users can leverage the dashboard to view information detail down to the seconds level and Enterprise Command Center dashboards can also display the time difference from the current time to the seconds level.

Example of a Time Stamp for Action Date



Oracle Enterprise Command Center Framework also considers the Oracle E-Business Suite user's time zone for displaying time-sensitive data. This capability improves information accuracy and reliability.

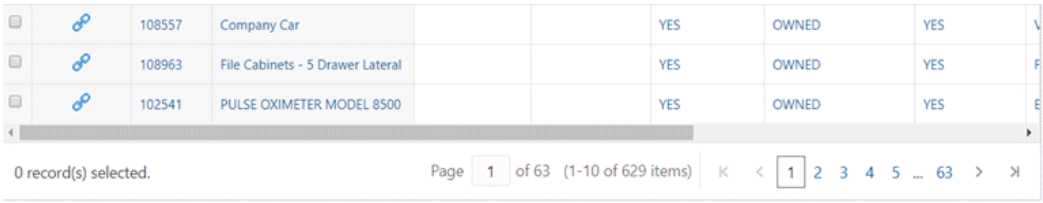
Common Component Controls

Pagination Controls

Components that display detailed data includes pagination controls that allow navigation between pages. The following components include pagination controls:

- Results Table
- Aggregated Table

Pagination Controls



Runtime Options

In some components, the runtime options includes several options that enable the business user to control how the data is displayed in the component. . Starting with V11, the runtime options support inline labels to improve form usability. The following

components include a runtime option:

- Tag Cloud
- Chart
- Search Component
- Available Refinements
- Summarization Bar
- Results Table
- Timeline View
- Aggregated Table
- Pivot View
- Results Grid

Tag Cloud

A tag cloud can be used if there are dimensions and multiple metrics available (based on the configuration). From the runtime option, the user can select a metric or dimension from the drop-down list to select the dimension or metric to be displayed.

Tag Cloud Runtime Options

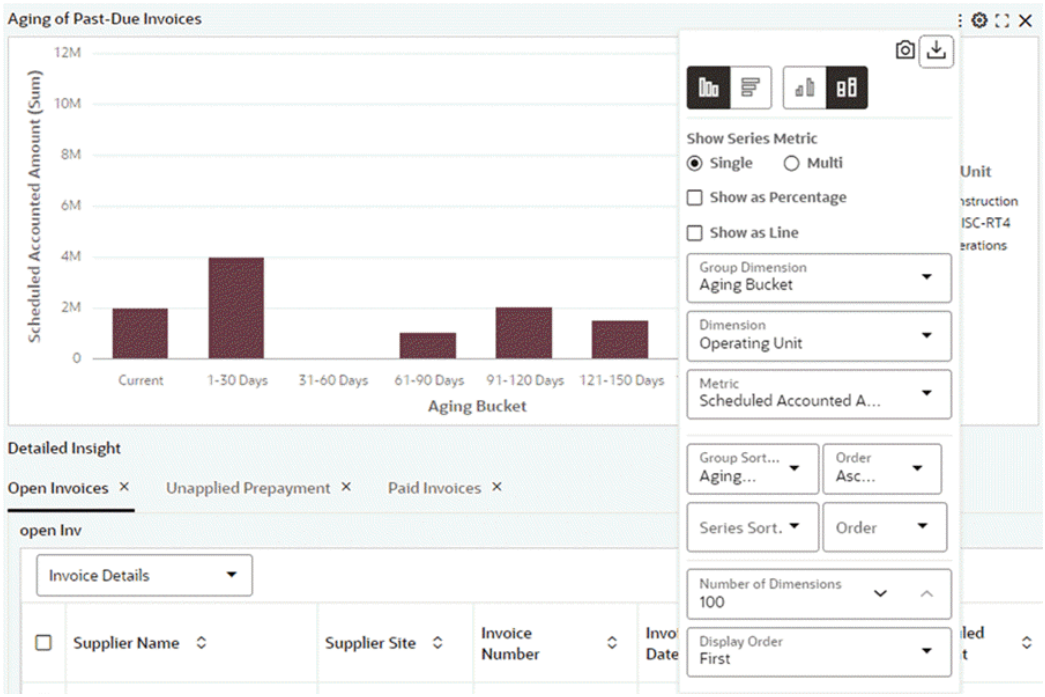
Chart

Configuration options include:

- A chart may be used if there are dimensions and multiple metrics available (based on the configuration). The user can select a metric or dimension from the drop-down list to select the dimension or metric to be displayed.
- Control the chart display either vertically or horizontally, stacked or unstacked.
- Control display of only one metric, or multiple metrics if more than metric is defined.

- Show the chart as a line instead of a bar.
- Show/hide refinement line (if configured).
- Change chart sorting as runtime (if enabled).

Chart Runtime Options

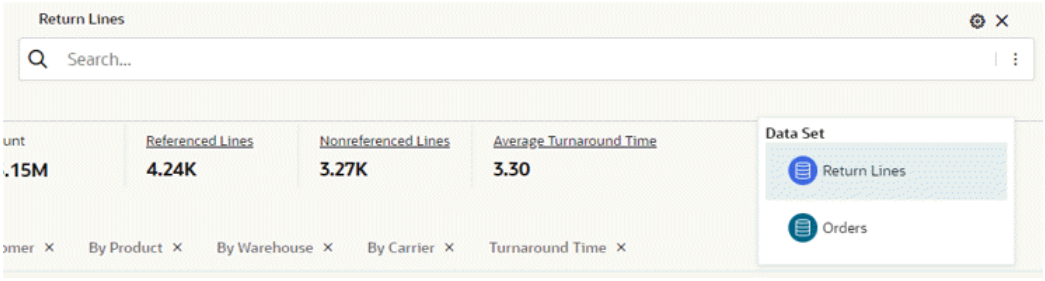


Search Component

A user can switch among the data sets to search for values from each data set and apply as filters.

The name of the data set appears in the component title.

Search Component Runtime Options

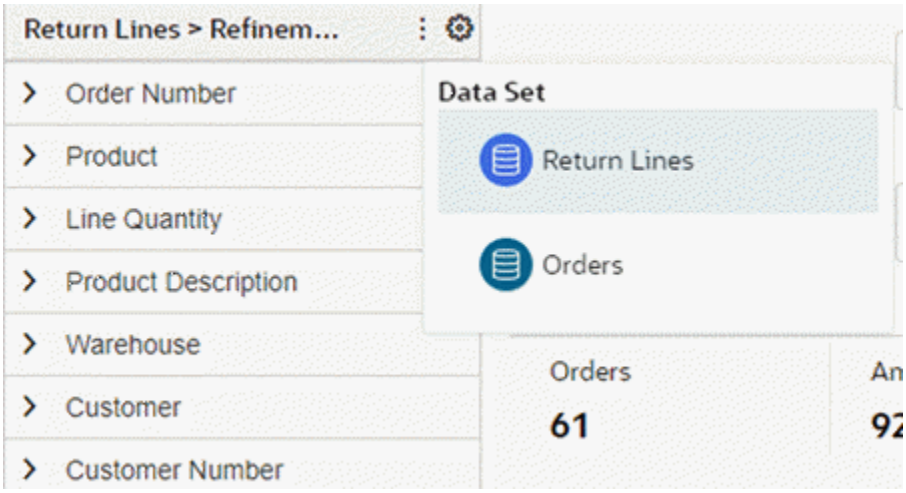


Available Refinements

A user can switch among the data sets to apply attribute values as filters.

The name of the data set appears in the component title.

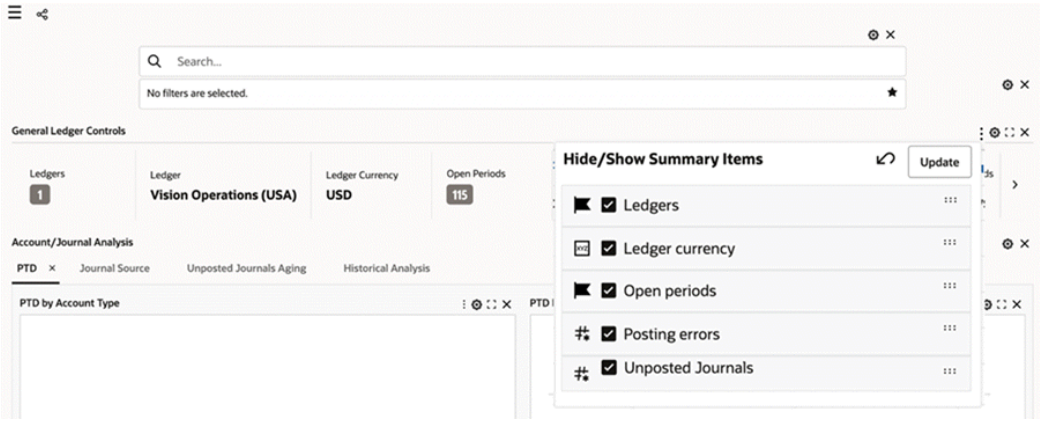
Available Refinements Runtime Options



Summarization Bar

Beginning with V10, users can select, deselect, or reorder summary bar items to be displayed through runtime options.

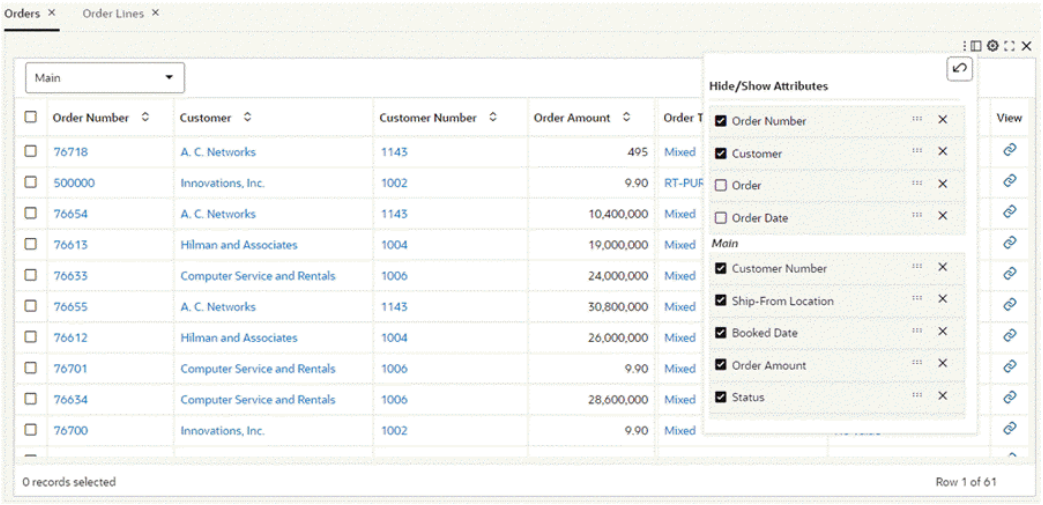
Summarization Bar Runtime Option



Results Table

Beginning with V10, users can hide, show, or reorder the result table attributes they want to view through runtime options.

Results Table Runtime Options

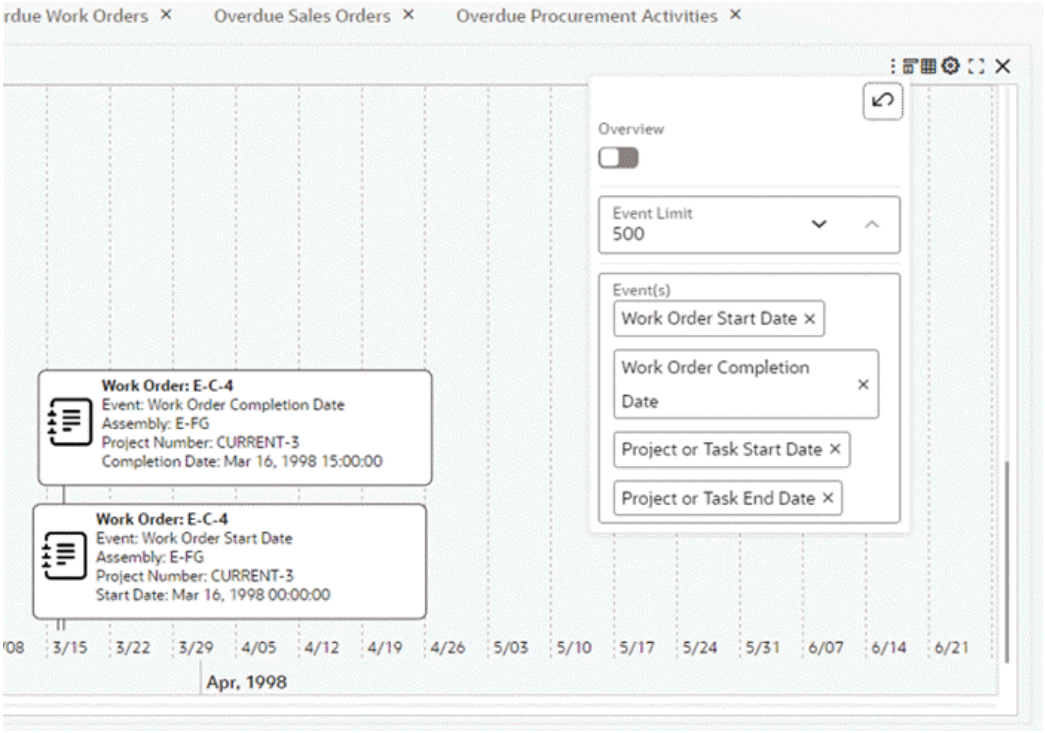


Timeline View

End users can do the following through runtime options:

- Add or remove the timeline events
- Enable or disable the overview
- Set the event limit

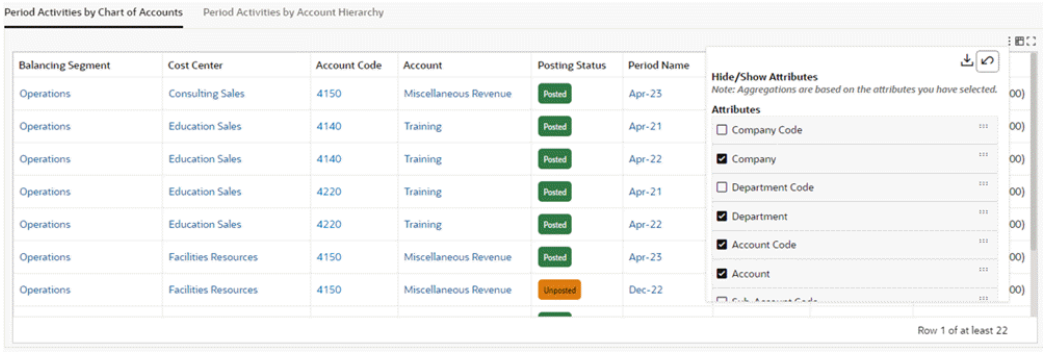
Timeline View Runtime Options



Aggregated Table

Users can hide, show, and rearrange the attributes and metrics using the runtime options.

Aggregated Table Runtime Options

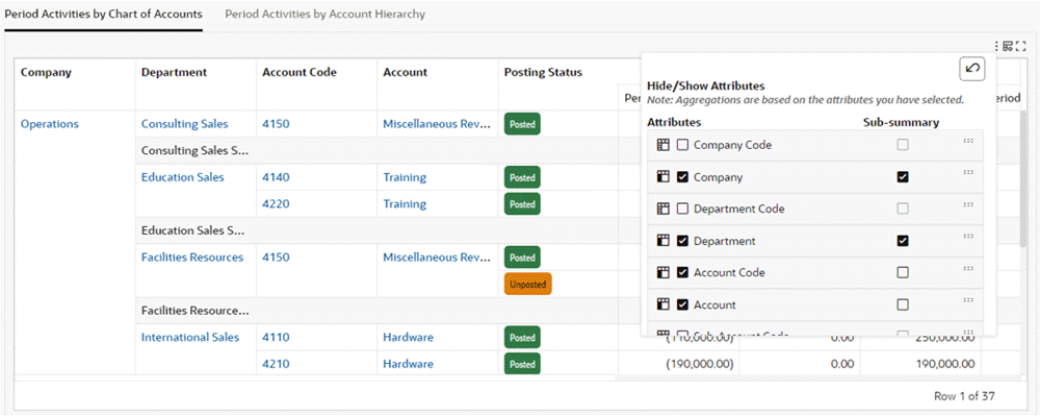


Pivot View

Users can hide, show, and rearrange the attributes and metrics using the runtime options.

Users can also enable or disable the subsummary for an attribute.

Pivot View Runtime Options

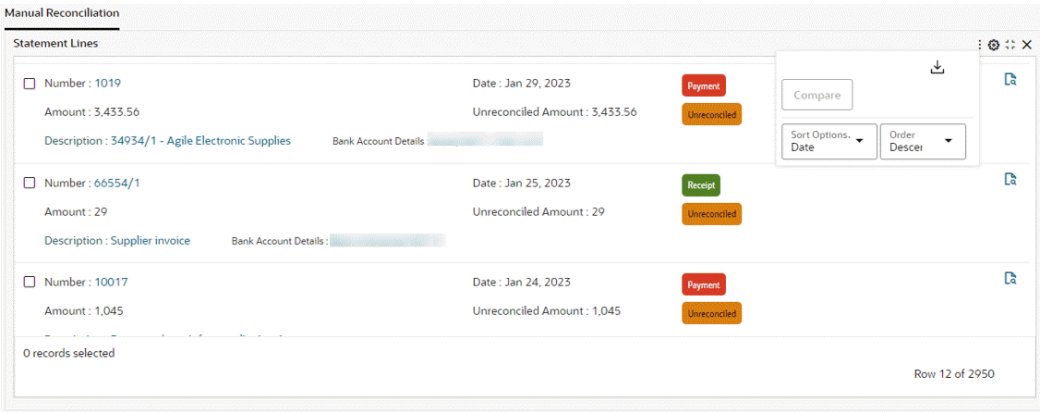


Results Grid

Users can compare multiple records (if enabled).

Users can reorder the appearance of records (If enabled).

Results Grid Runtime Options



Retain Selected Refinements across Dashboards

Beginning with V10, designers can configure a component to preserve selected refinements across dashboards. With this feature enabled, when the end user scrolls away from a page (via a component) and then returns to it, the selected refinements will be preserved. This new feature spans across all components that support hyperlinks: Results Grid, Search, Results Table, Aggregate Table, Aggregate Grid, Summarization bar, and Diagram. Designers can choose to preserve selected refinements in the component's configuration by selecting the **Retain Existing Refinement State on**

Source Page checkbox.

ECC Hyperlinks

Hyperlinks in ECC are primarily used to facilitate navigation by providing links that redirect users to specific destinations. Integrating hyperlinks into ECC dashboards greatly enhances interactivity, leading to an improved user experience.

Hyperlinks in ECC are supported in the summary bar (metric item), aggregated table, result table, grid, and diagram.

Depending on the business case, the structure of the hyperlink differs. There are essentially three distinct hyperlink structures within ECC.

Navigating to a Form

Follow these steps to enable links for an Oracle E-Business Suite window in Oracle Forms.

The syntax for the link is:

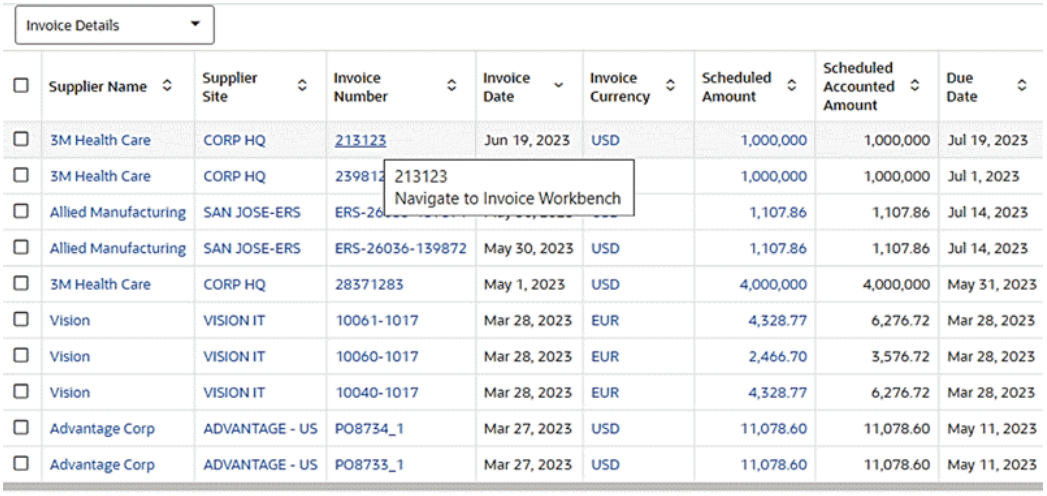
```
/ecc/ebs/formsCaller?functionCode={form function}
```

Where `functionCode` is the form function associated with the form.

Use `Parameter` to pass an argument to the form function.

For example: To pay an open invoice, the payables manager wants to click on the invoice number of the result table component to navigate to the Invoice Workbench form. The hyperlink needs to be configured on the attribute (invoice number) to facilitate seamless navigation to the Invoice Workbench form.

Example of a Results Table with a Link to the Invoice Workbench



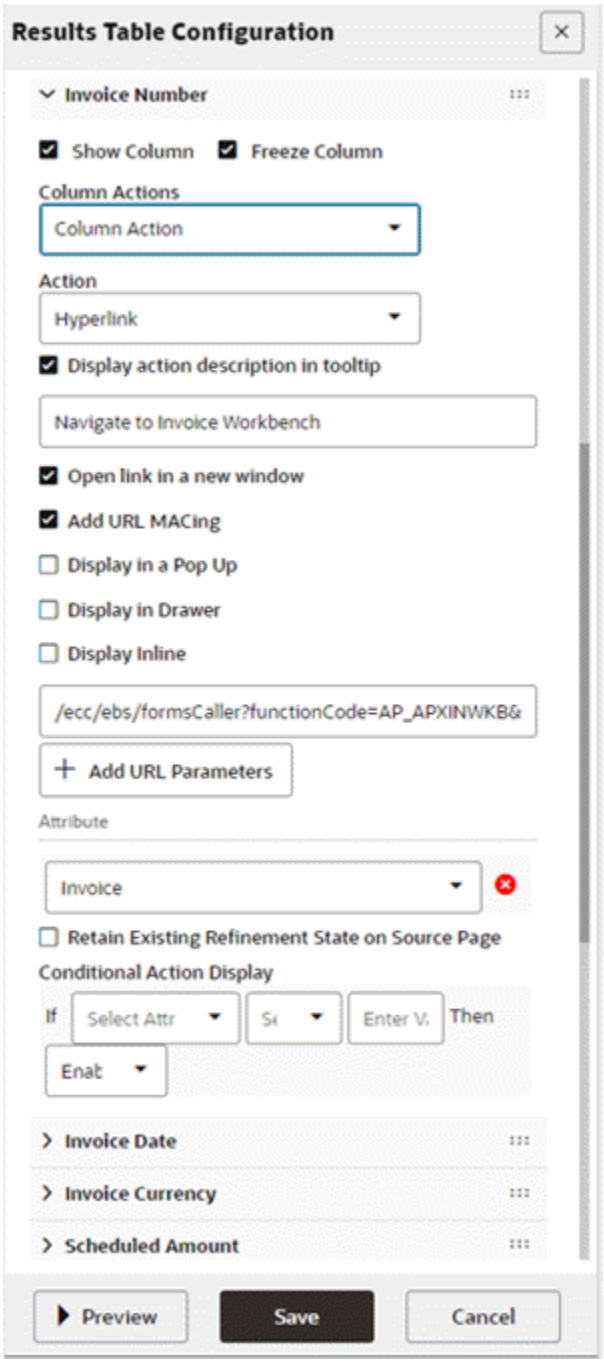
The steps to configure this link are:

1. For the Invoice Number attribute, Set the column action to "Column Action."
2. Set the action as "Hyperlink."
3. Enable **Display action description in tooltip**.
4. Provide business-friendly text, such as "Navigate to Invoice Workbench," for the tooltip.
5. Select **Open Link in a new window** to open the form.
6. Select **Add URL MACing**.
7. Set the Hyperlink URL:

   `/ecc/ebs/formsCaller?functionCode=AP_APXINWKB&INVOICE_ID={0}`

   where `AP_APXINWKB` is the form function of the Invoice Workbench.
8. Set the URL Parameter to "Invoice." The INVOICE_ID that is used in the URL is the attribute key for the attribute `Invoice`.

Example of the Results Table Configuration for the Link to the Invoice Workbench



Navigating to an Oracle Application Framework Page

Follow these steps to enable links for Oracle Application Framework (OAF) pages.

The syntax for the link is:

```
/OA_HTML/OA.jsp?OAFunc={OA Page}
```

Where `OAFunc` is the FND function associated with OA Page.

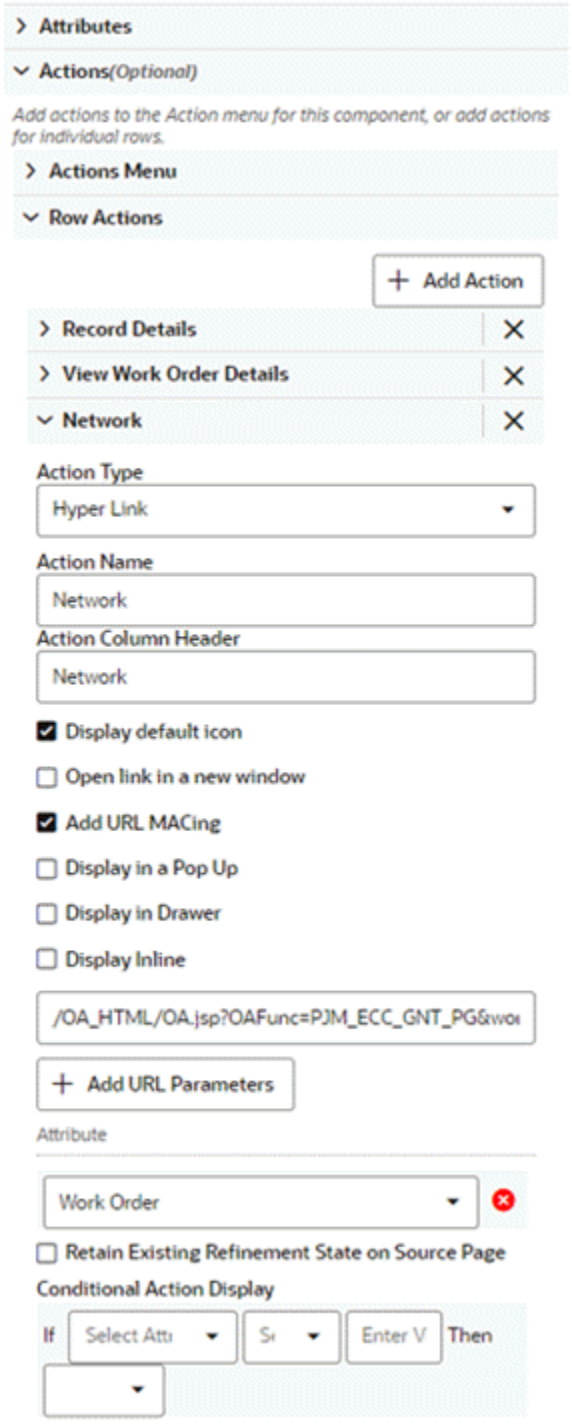
`Parameter` is optional. Use it to apply a refinement on the target OAF page.

For example: While reviewing the "Work Order Schedule exceptions" in an results table, a Project Manufacturing user wants to review the genealogy of a specific work order. A hyperlink thus needs to be configured as a row action. It will be called "Network," as shown below.

The steps to configure this link are:

1. Inside the **Action** accordion of Row Actions, add a Row Action.
2. Define the action type as Hyper Link.
3. Define the Action Name "Network."
4. Define Action Column header "Network."
5. Select **Display default icon** to prefix the action with an icon
6. Select **Add URL MACing**.
7. Define the Hyperlink URL:
   ```
   /OA_HTML/OA.jsp?OAFunc=PJM_ECC_GNT_PG&workOrder={0}
   ```
8. Set the URL parameter to "Work Order."

Example of the Results Table Configuration with a Link to a Work Order Page



Navigating to an Oracle Enterprise Command Center Framework Page using Deep Links

A deep link in ECC serves as a specialized hyperlink that enables seamless navigation

to another dashboard while preserving pre-set filters. This functionality is pivotal for maintaining context within a specific business area. Deep links accommodate various ECC refinements and even allow the configuration of hidden filters within chosen refinements. They are especially valuable when users need to retain filter context across dashboards to carry out related tasks by offering a smooth transition to different dashboards for these activities.

Configuring Deep Links

Oracle Enterprise Command Center Framework supports the following filters as part of deep links:

- Selection filter
- Range filter
- Search filter
- Clear filter

The generic syntax for configuring a deep link is as follows:

```
eccDeepLink=[{"dataset_key": [<filters>]}]
```

For configuring a selection filter, the syntax is

```
eccDeepLink=[{"dataset_key":[{"attributeKey":"<attribute_key>","values":
["<values>"],"operator":"=="}]}]
```

Properties for a Selection Filter

Property	Description
attributeKey	The attribute key of the data set.
values	The attribute value to use for the refinement.
operator	One of the operations to apply on the data set attribute: `==, null, !null, !=, >,<, >=,<=`
hide	Optional. Set this to `true` to hide the selected attribute on the selected refinement UI.

An example of the syntax for configuring a range filter is:

```
eccDeepLink=[{" dataset_key ":[{"from":"2001-02-26","to":"2006-12-13","
attributeKey":"< attribute_key >"},{"from":"720","to":"90000","
attributeKey":"<attribute_key>"}]}]
```

Properties for a Range Filter

Property	Description
attributeKey	The attribute key of the data set.
from	The lower bound number or date. The date should be in canonical format.
to	The upper bound number or date. The date should be in canonical format.
hide	Optional. Set this to `true` to hide the selected attribute on the selected refinement UI.

An example of the syntax for configuring a search filter is:

```
eccDeepLink=[{"fa-asset":[{"search":["cip","computer"]}]}]
```

Properties for a Search Filter

Property	Description
search	Array of search keywords, comma-separated; every keyword should be at least three characters long.
hide	Optional. Set this to `true` to hide the selected attribute on the selected refinement UI.

The syntax to clear the filters in the target dashboard is

- To reset all the selected refinements: *`eccDeepLink=[]`*
- To reset filters from a data set: *`eccDeepLink=["<dataset_key>":{}]`*

Example of a Deep Link

For example, say in Project Manufacturing, the Project Manufacturing Execution Genealogy diagram is utilized for tracking detailed project manufacturing activities. When dealing with a particular work order, it's essential to review the work order's specific timeline to grasp its start and completion. The solution here is to implement a hyperlink action for the work order attribute, enabling direct access to the work order's dedicated timeline without the need for cumbersome manual filtering.

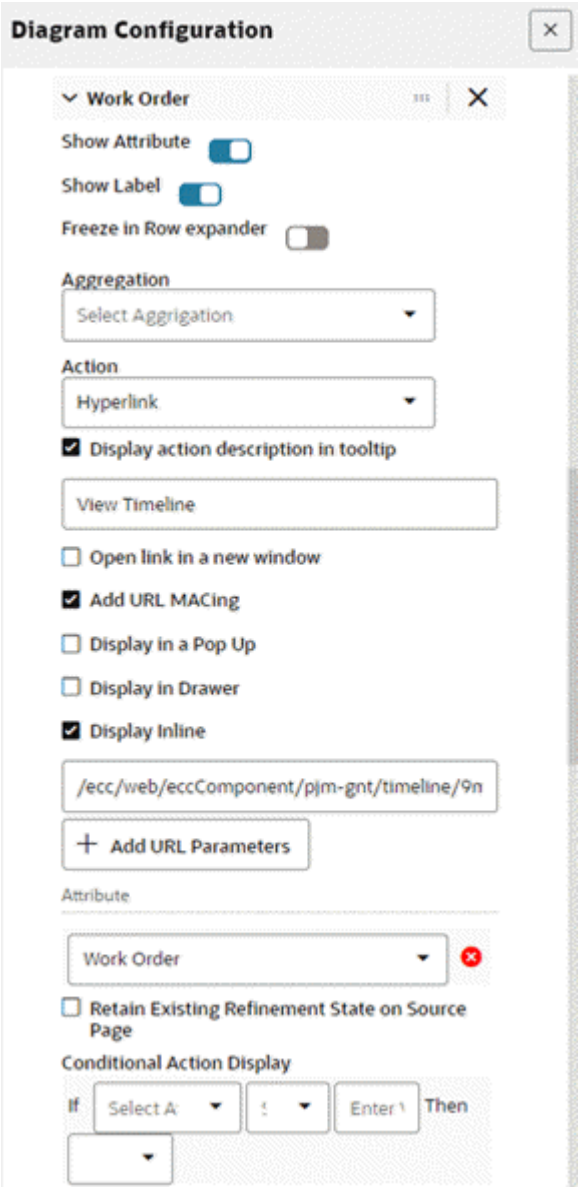
The steps to configure the link are:

1. Set the Action Type as "Hyperlink" for Work Order attribute.
2. Select "Display action description in tooltip"
3. Set the tooltip description with a business-friendly name "View Timeline"
4. Select "Add URL MACing"
5. Select "Display Inline" as the preferred medium to access timeline. Timeline is a component, rather an ECC page. Hence, accessing it using inline view makes more sense as it will allow the user to see the timeline while remaining in context of the dashboard
6. Set the URL as below:

   ```
   /ecc/web/eccComponent/pjm-gnt/timeline/9mo5xgsyt1r?eccDeepLink=
   [{"pjm-gnt-wo":[{"attributeKey":"JOB_NAME","values":["{0}"],"
   operator":"=="}]}]
   ```

   In cases where a deep link is required to provide direct navigation to a component within the target page, the URL path is constructed with a page URL followed by the unique component ID, such as "/ecc/web/eccComponent/pjm-gnt/timeline/9mo5xgsyt1r."

Example of a Diagram Configuration with a Link for a Work Order's Timeline



However, in certain situations where the entire page's content needs to be displayed instead of just a specific component, a different URL structure is employed. For instance, when accessing an associate's expenses, the entire expense analysis dashboard associated with that associate is presented, and the URL takes the form of "/ecc/web/eccapp/ap-emp-exp/ap-exp-ana."

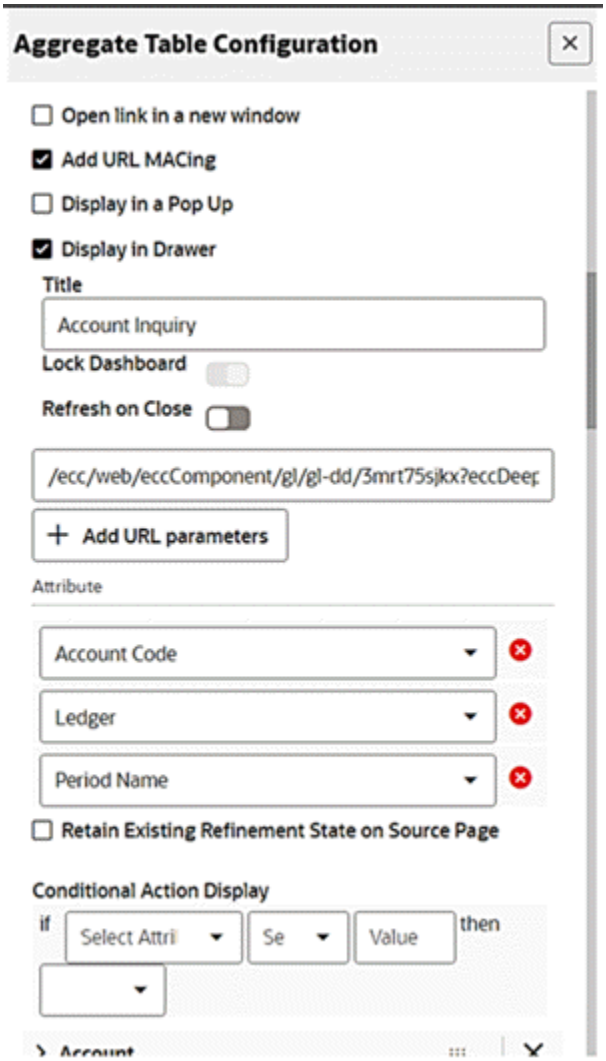
In this alternate structure, "eccApp" is used instead of "eccComponent," and there is no inclusion of a specific component ID since the objective is to present the entire page's content.

7. Set the URL parameter to "Work Order".

An example of a deep link with multiple parameters is:

```
/ecc/web/eccComponent/gl/gl-dd/3mrt75sjkx?eccDeepLink=[{"gl-aa":
[{"attributeKey":"GL_ACCOUNT_CODE","values":["{0}"],"operator":"=="},
{"attributeKey":"LEDGER","values":["{1}"],"operator":"=="},
{"attributeKey":"PERIOD_NAME","values":["{2}"],"operator":"=="}]}]
```

Example of an Aggregate Table Configuration with Multiple Parameters



In scenarios where a deep link incorporates multiple attributes, these attributes are assigned sequential numbers for their values. It is crucial that the URL parameters adhere to the exact same sequence used in the URL.

An example of a deep link with a Hide parameter is:

```
/ecc/web/eccapp/ap-emp-exp/ap-exp-ana?eccDeepLink=[{"ap-emp-exp":
[{"attributeKey":"EMPLOYEE_ID","values":["{0}"],"operator":"==","hide":
true}]}]
```

In certain situations, concealing the applied refinement from end user becomes necessary. In such instances, the "hide" parameter proves to be valuable.

Custom Label

Introduced in ECC V12, the Custom Label feature enhances effective communication by enabling users to align aggregated data with their organization's or team's internal language and conventions. Additionally, it enhances flexibility and adaptability by accommodating diverse user requirements, ensuring that visualizations are relevant and meaningful to their specific use cases.

UI Elements in which Custom Label is Effective

Component	UI Elements
Summary Bar	Flag pop-up metric title
Chart	Axis title
	Tooltip
	Default chart title
	Chart legend
	Runtime option
	Export
	Cascaded chart
Aggregated Table (Pivot view)	Column headers
	Runtime option
	Export
Tag Cloud	Default component title
	Runtime option

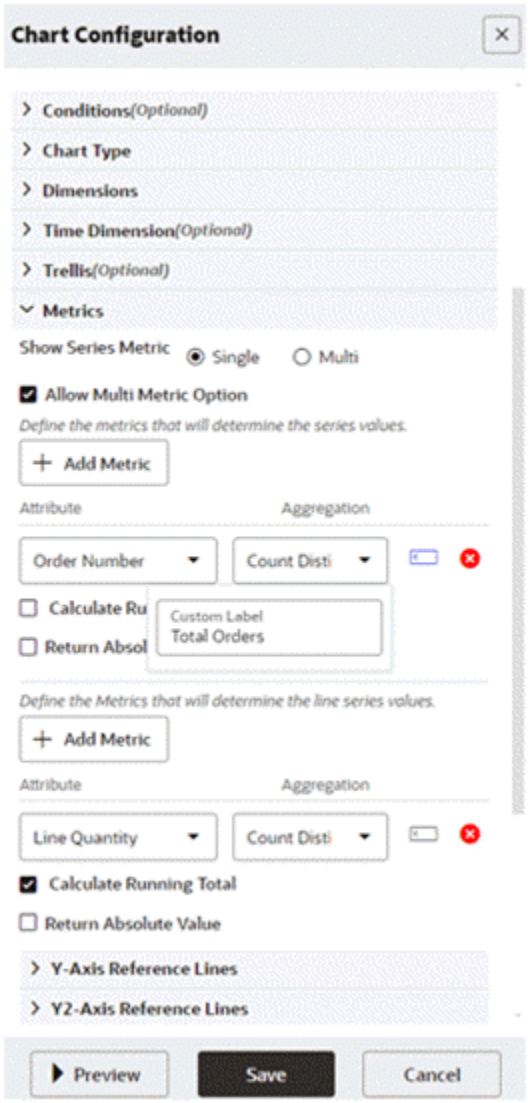
Component	UI Elements
Network Diagram	Node
	Tooltip
	Row expander
	Export
	Show detail
	Compare
Grid	Cell label
	Export

An example of a custom label for a chart could be one defined as follows:

- Chart Type: Bar/Line
- Bar Metric: Order Number (Count Distinct)
- Custom Label for Bar Metric: Total Orders
- Line Metric: Running Total (Line Quantity (Count Distinct))

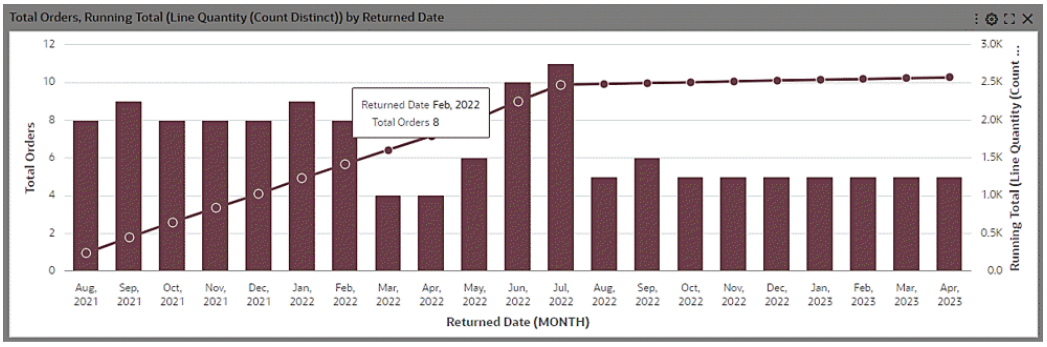
> **Note:** For the line metric, a custom label is not defined. Hence, the icon is not active. Note how the custom label icon has turned blue for the Bar Metric, indicating that a custom label is set.

Configuration Options for a Chart with a Custom Label



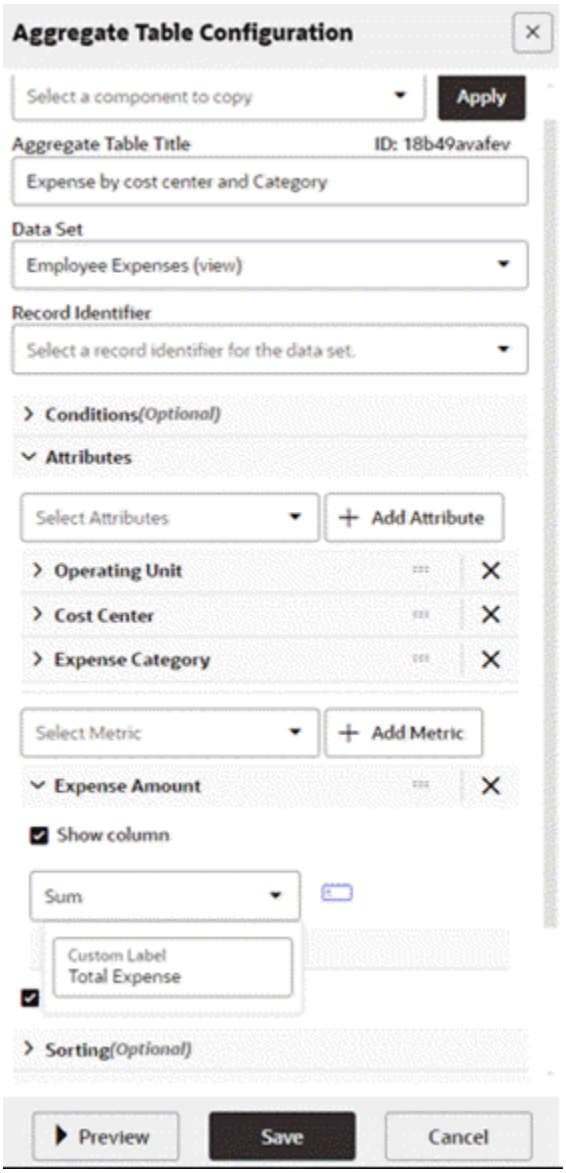
Note the effect of a custom label in the axis title, tooltip, and chart title.

Preview of a Custom Label in a Chart



For an example with an Aggregated Table, a custom label could be defined on the metric "Expense Amount (Sum)".

Example of Aggregated Table Configuration with a Custom Label



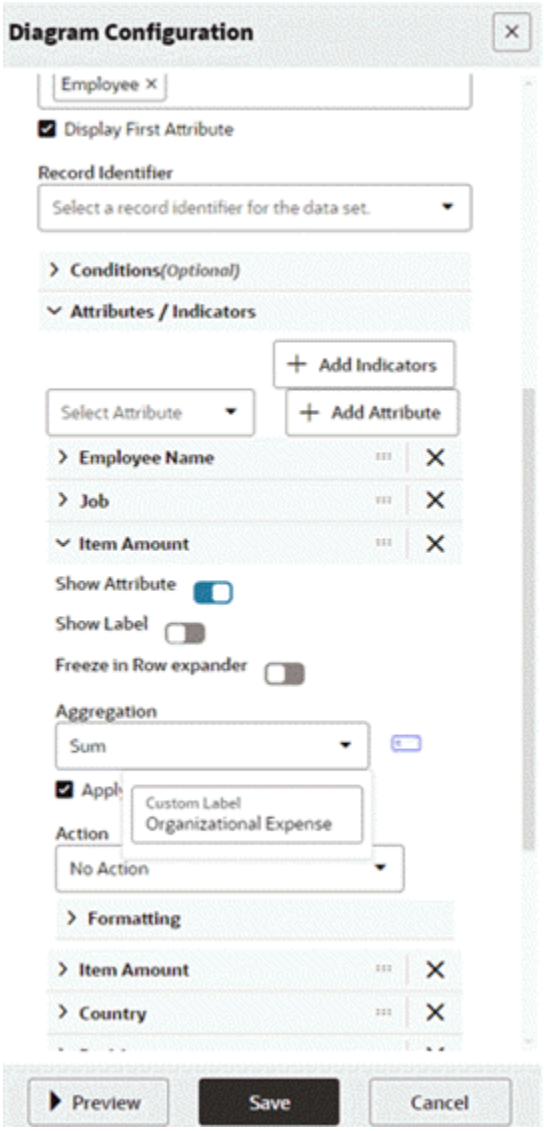
A preview of the aggregated table shows the effect on the runtime context menu and column headers.

Example of an Aggregated Table with a Custom Label

The following is an example of a network diagram (Hierarchical Query).

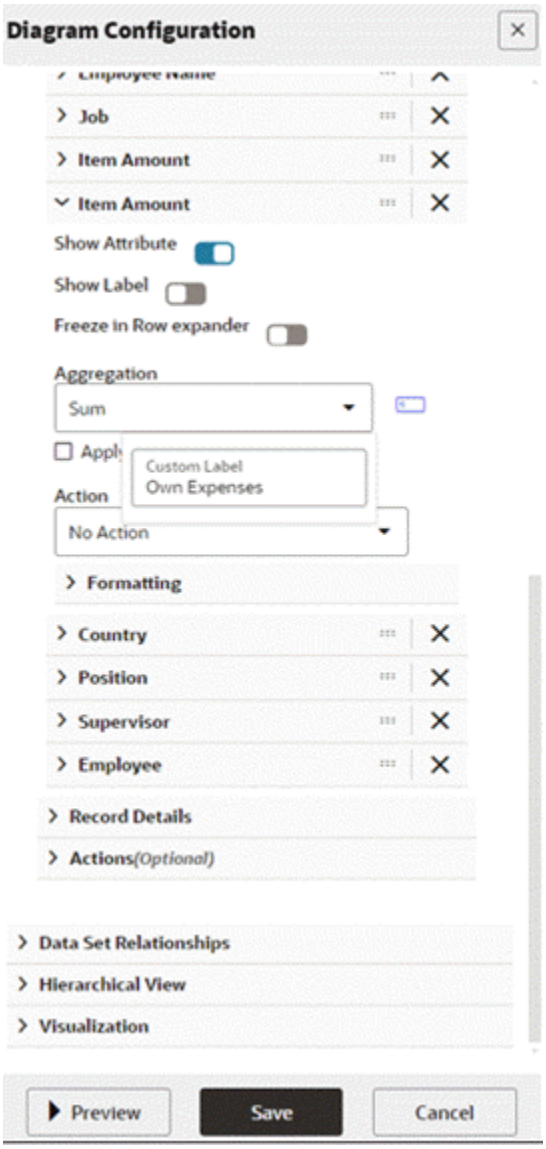
The custom label "Organization Expense" is set where hierarchical aggregation is enabled:

Diagram Configuration with a Custom Label where Hierarchical Aggregation is Enabled



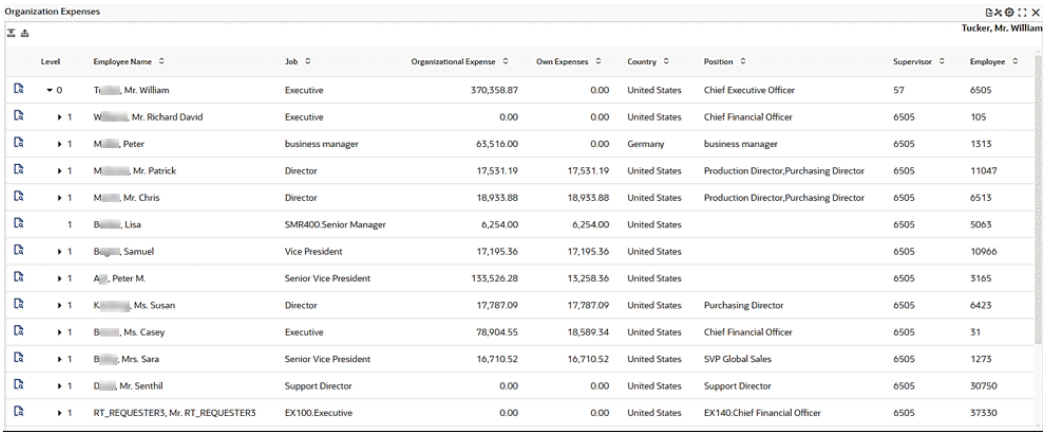
The custom label "Own Expense" is set where hierarchical aggregation is not enabled.

Diagram Configuration with a Custom Label where Hierarchical Aggregation is Disabled



In the Row Expander view of the diagram, the custom labels "Organizational Expense" and "Own Expenses" are used as column headers.

Example of Custom Labels in a Diagram's Row Expander View



Level	Employee Name	Job	Organizational Expense	Own Expenses	Country	Position	Supervisor	Employee
0	T[redacted], Mr. William	Executive	370,358.87	0.00	United States	Chief Executive Officer	57	6505
1	W[redacted], Mr. Richard David	Executive	0.00	0.00	United States	Chief Financial Officer	6505	105
1	M[redacted], Peter	business manager	63,516.00	0.00	Germany	business manager	6505	1313
1	M[redacted], Mr. Patrick	Director	17,531.19	17,531.19	United States	Production Director,Purchasing Director	6505	11047
1	M[redacted], Mr. Chris	Director	18,933.88	18,933.88	United States	Production Director,Purchasing Director	6505	6513
1	B[redacted], Lisa	SMR400.Senior Manager	6,254.00	6,254.00	United States		6505	5063
1	B[redacted], Samuel	Vice President	17,195.36	17,195.36	United States		6505	10966
1	A[redacted], Peter M.	Senior Vice President	133,526.28	13,258.36	United States		6505	3165
1	K[redacted], Ms. Susan	Director	17,787.09	17,787.09	United States	Purchasing Director	6505	6423
1	B[redacted], Ms. Casey	Executive	78,904.55	18,589.34	United States	Chief Financial Officer	6505	31
1	B[redacted], Mrs. Sara	Senior Vice President	16,710.52	16,710.52	United States	SVP Global Sales	6505	1273
1	D[redacted], Mr. Senthil	Support Director	0.00	0.00	United States	Support Director	6505	30750
1	RT_REQUESTER3, Mr. RT_REQUESTER3	EX100.Executive	0.00	0.00	United States	EX140.Chief Financial Officer	6505	37330

Personalization

Introduced in V10, User Personalization consists of two types:

- End User Personalization, formerly known as User Preferences
- Power User Personalization

End User Personalization

End User Personalization allows end users to make changes in the dashboard that will suit their preferences. End users can do the following:

- Enhance business user experience by editing components in the dashboard
- User changes are preserved across the session.
- The component displays according to each user's preferences.

  End user personalization is available across visualization components such as Summary Bar, Tag Cloud and Charts, and across detailed insights and aggregated components like Results Table/ Timeline, Aggregate Table/ Pivot and Grid/ Aggregated Grid. Each component locally has a reset option under runtime options that the user can use to return to the default view of the component. End user personalization can be reset either at the Component level or at the Page level. Below are examples of end user personalization done at the Component level.

Chart

Users can personalize the chart to suit their needs by flipping the chart layout or using a different metric/dimension.

Tag Cloud

User can personalize the Tag Cloud to suit their needs by using different metric/dimension.

Summarization Bar

User can personalize the Summarization Bar to suit their needs by reordering summary items, showing/hiding summary items from the view.

Results Table

User can personalize the Results Table to suit their needs by reordering attributes, showing/hiding attributes from the view.

Aggregate Table/Pivot

User can personalize the Aggregate Table/Pivot to suit their needs by reordering attributes, showing/hiding attributes, enabling/hiding sub-summary from the view.

Timeline

User can personalize the Timeline to suit their needs by changing the event limit, or choosing the types of events shown the view.

Reset End User Personalization at Page Level

End User Personalization can be reset at Page level, this would reset End User Personalization for all components in the dashboard, as this is performed at the Page level.

Power User Personalization

Power users can modify the dashboard to tailor it based on their business requirements.

Examples:

- A Credit Manager wants to track top past-due customers based on their profile class so that they can revisit customers' credit ratings and manage or update the payment terms.
- A Payables Accountant wants to track the cash outflow for invoice payments per bank account so that they can analyze payments trend.

A Power User has:

- Same designer experience as an administrator user.
- Enabled at the user level using the profile option "FND: ECC Power User Enabled"

- All personalizations are saved without any deployment.
- No access to the Enterprise Command Center Framework Administration UI.

Beginning with V11, an "i" icon on the dashboard is available. Clicking this icon opens a window with additional information as well as a **Personalize** button to enable personalization and an **Exit** button to exit personalization.

Examples of Power User Capabilities

Can Perform	Cannot Perform
Add a new component	Create a new data set
Modify an existing component	Create a new dashboard
Delete a component	Configure a component from a data set that is not part of the application datasets
Change dashboard layout	Create a public saved search

The following table summarizes the differences between the two types of personalization.

End User Personalization versus Power User Personalization

End User Personalization	Power User Personalization
All business users	Power users only
Runtime changes	Configuration changes (components)
Implicit saving as user utilizes run-time options	Explicitly saving for configuration changes
Involves no efforts from the user	Involves efforts from the user

Power user personalizations are preserved during patching. A user is notified when a new version is available. Users can apply the new version and lose all the personalizations, or they can cancel and keep the personalized version.

Personalizing a Page as a Power User

1. When personalization is enabled for a user (that is, the FND:ECC Power User Enabled profile option is enabled at the user level), a new icon placed inside the (i) icon to indicate that user can personalize the dashboard. The power user can do so by clicking on the Edit Dashboard Icon. This icon is black by default.

2. Once the user clicks on the icon, the icon color changes from black to blue, and the page is enabled for editing.

3. When the user applies a personalization on the dashboard, the icon becomes red, indicating that this page is personalized.

4. When the user hovers over the icon, a tooltip, "Personalized Dashboard," is displayed to indicate that the dashboard is personalized.

5. The user can reset the personalization by clicking on the Reset icon at the dashboard level.

6. When user enables a personalization on a page, and applies personalization on one of the existing components in the dashboard, or adds a new component, a new icon is displayed at the component level identifying that the component is personalized (this icon is visible only in personalization mode).

7. When the user hovers over the icon, the "Personalized" tooltip is displayed to indicate that the component is personalized.

8. A notification message is shown inside the information icon and has options to accept or cancel options. This icon has a red dot to indicate a new notification is present for the user.

    1. Once the power user applies the change, the red dot will disappear and the dashboard will be the latest version from the administrator.

    2. If the power user selects Cancel, then the icon is closed. If the power user opens the "i" icon again, they can see the message again inside it.

    3. At any time, the power user can click on the Reset icon to go to the latest version from the administrator.

    4. The power user can use the scrollbar to see all the data sets.

"Copy Component" Feature for Power Users

The Copy component feature is added in V11. Power users can copy any component in

a page that belongs to the same application.

To copy a component: In the Configuration window, a power user can:

1. Click on the Copy Component field to select which component to copy and from which page to copy.

2. Click Apply.

The source component's configuration will be copied over, and the power user can choose to either save the configuration as is, or choose to edit the component further and then save the configuration.

All configuration and all runtime options will be copied from source component in the target component.

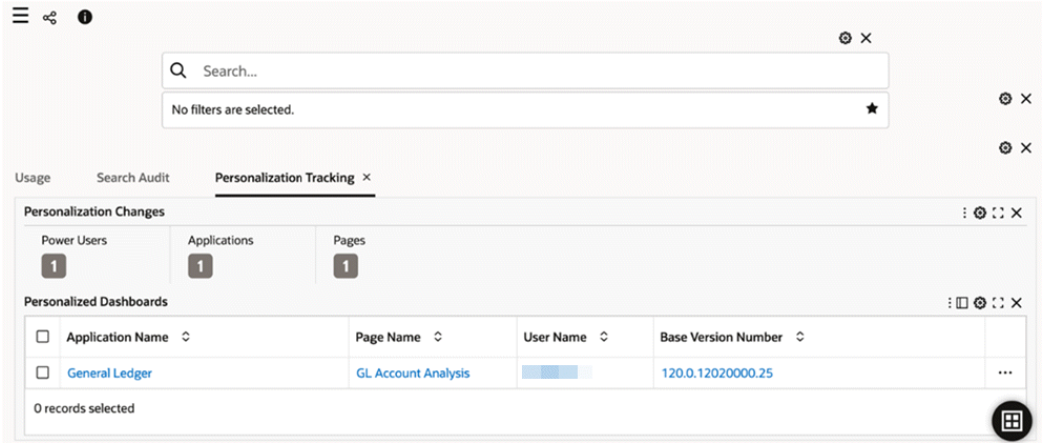
Only 1:1 mapping of components is possible; that is, from a results table component the Copy Component list will display only results table components across dashboards in that application to copy. For example: if you are copying within a Chart component: The list will display only chart components across dashboards within the current application.

Personalization Tracking

Beginning with V11, administrators can track and revert personalization changes made by power users.

Admins can revert all personalization made by power users at the page level to the site level dashboard or revert individual component updates.

Personalization Tracking Tab



Share Personalization

Introduced in V12, Share Personalization functionality presents enhanced user engagement and fosters improved collaboration and communication. It enables users to share personalized dashboards seamlessly across an Oracle E-Business Suite responsibility.

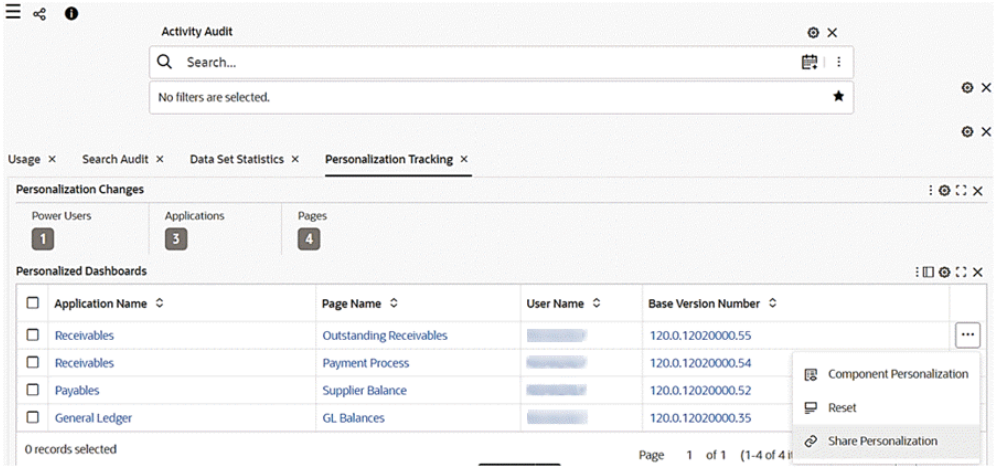
Share Personalization Process Example

1. A power user has requested an administrator to share their personalization with everyone using their responsibility.

   The administrator visits the activity audit dashboard and filters by Page name and username to find the personalization detail.

   The administrator accesses the corresponding row action "Share Personalization."

Share Personalizations Action Selected on Personalizations Tracking Tab

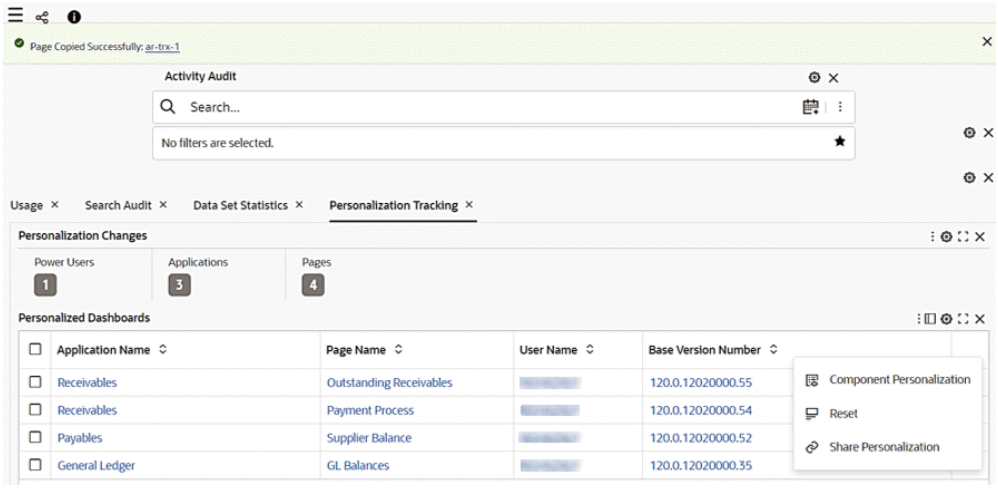


2. As soon as share personalization is clicked, the page is copied.

   A message with the page key provided upon successful copy of page appears upon the successful copying of page.

   The message disappears only when user explicitly click on close button corresponding to the message.

Example of "Page Copied Successfully" Message



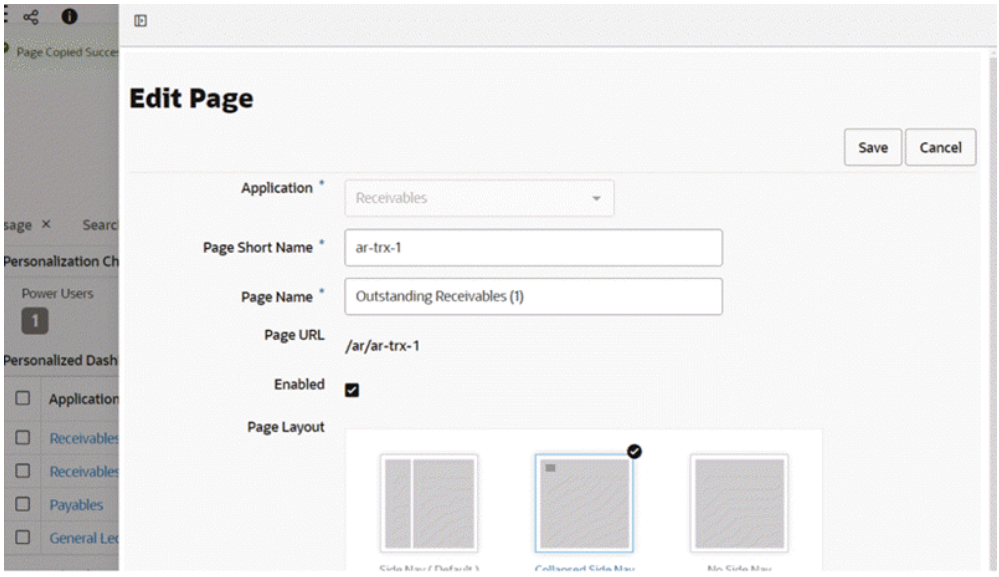
3. Upon a user clicking the hyperlink, a drawer opens with page detail of the copied page and the user can make amendments and save the changes.

   The naming convention of the copied page is as follows:

   - Page short name: `<existing name> <-> <personalized>-1`
   - Page Name: `<existing name> <space> <Personalized (1)>`

   Note that the first copy has the number *1*, and the nth copy would have the number *n*.

Edit Page for the Copied Page

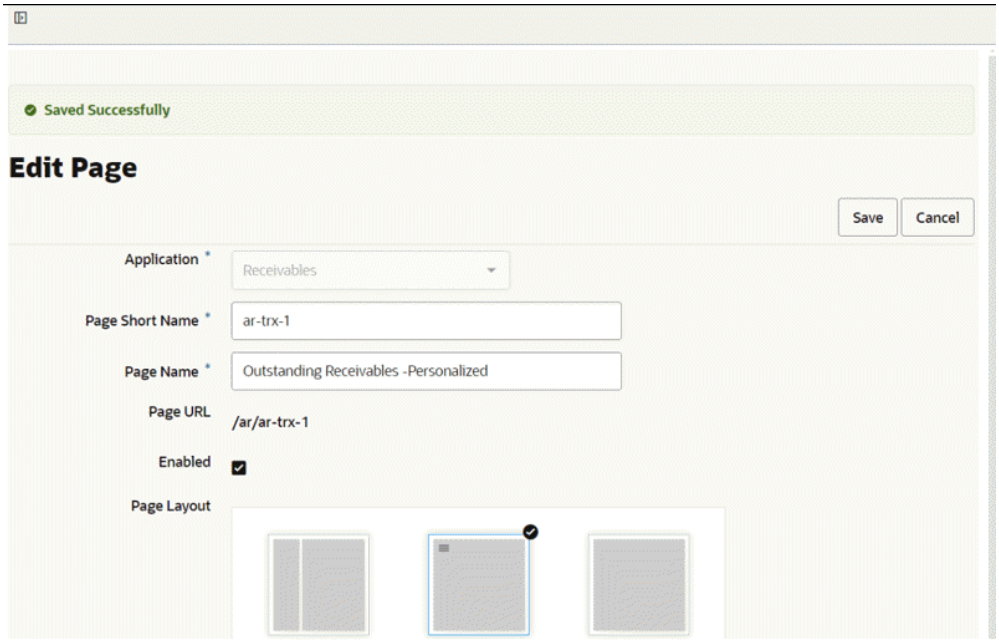


4. If the user amends name, key , or orientation and clicks **Save**, a message appears inside the drawer indicating that the changes are saved.

   The drawer is closed automatically.

   If the user clicks **Cancel**, the drawer is closed immediately.

"Saved Successfully" Message for an Updated Page



5. The administrator then continues with the rest of the process, such as creating new FND function and assigning a permission set in an Oracle E-Business Suite environment to complete the process.

Once the personalization is shared, if any of the users using the shared version of personalized wants to use the original version of the dashboard then they need to collaborate with the administrator. In a situation where a few users want the personalized version of the dashboard and a few want the shipped version of the dashboard, the administrator needs to create different responsibilities to handle those scenarios.

> **Important:** When a newer product version is adopted, there will be no automatic notification since the shared personalization page is essentially a copied page. Therefore, any communication pertaining to the newer shipped version must be handled offline between the administrator and the business users. It is essential to ensure that all relevant parties are informed and updated regarding any changes or updates to the product version to maintain effective collaboration and workflow management.

Collaboration

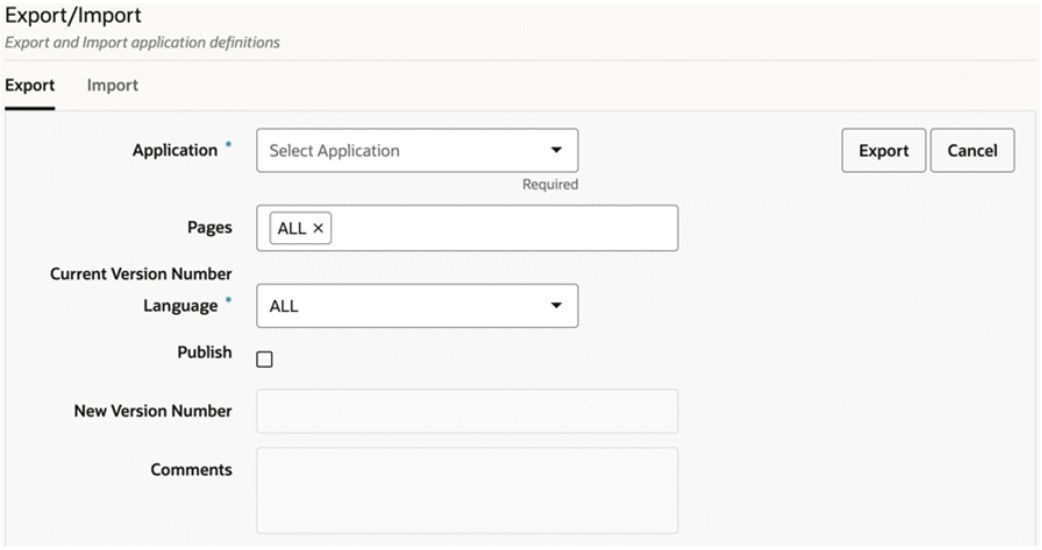
Collaboration: Export

Beginning with V11, an administrator can export a subset of pages when exporting the application. This flexibility allows the administrator to target and update only the pages that they intend to modify is targeting without affecting other pages within the destination application during the import process. Previously, an update to a single page necessitated exporting the entire application, with unintended updates to other pages in the process.

All pages are initially selected by default; administrators can deselect and choose specific pages.

Associated views of exported data sets are automatically included in the export.

Export/Import Page



Collaboration: Import

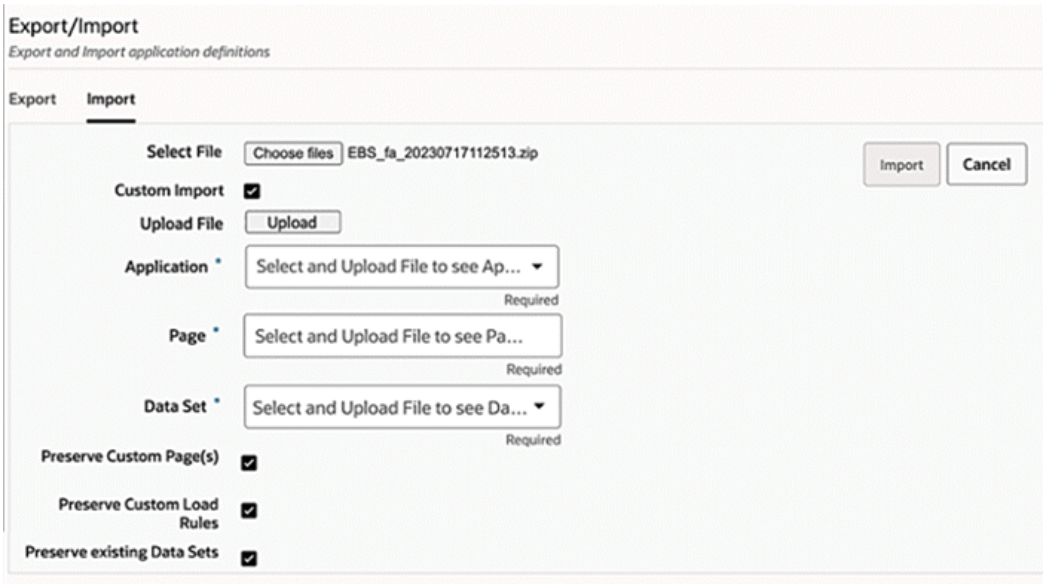
Introduced in V12, the Import feature:

- Allows developers to merge their Applications/Pages easily without having to override/disrupt other people's changes.
- Provides users the ability to import a subset of pages and data sets within an application.
- Provides users the ability to choose to preserve data sets.

Users must enable the **Custom Import** checkbox to see a more detailed selection choice, which allows them to import a subsection of pages or data sets, within the same or different application. It also allows them to choose to preserve existing data sets.

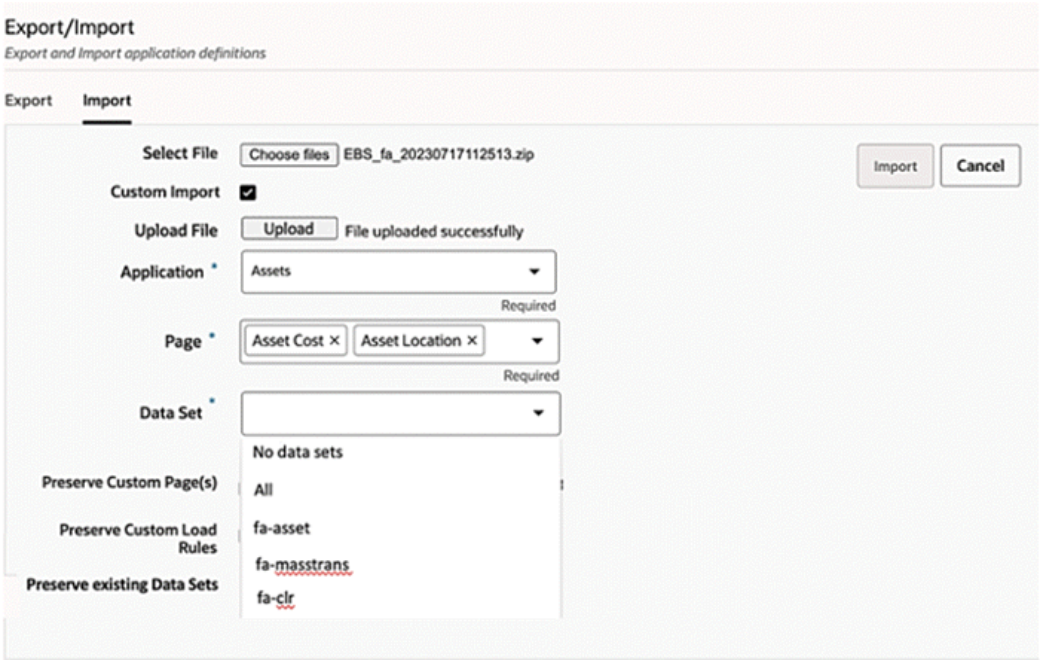
Users must select the necessary ZIP file, upload it, and then choose the application into which to import. They must then choose the respective pages and data sets, and then click on **Import**.

Import Tab on the Export/Import Page



For example, a user could choose to import the Asset Cost and Asset Location pages for the Assets application.

Example of Importing Specific Pages for an Application



Collaboration: Move/Copy Pages

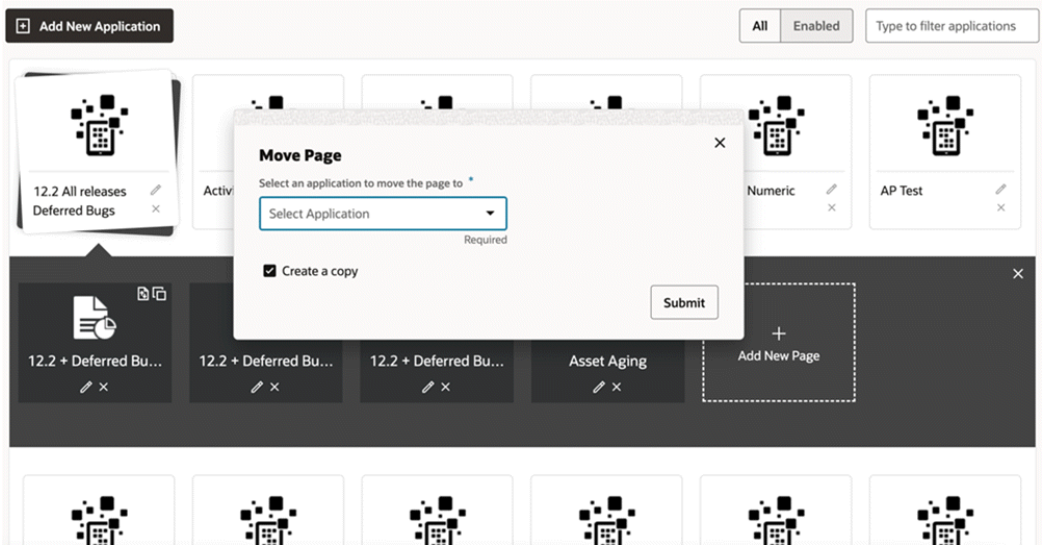
Beginning with V11, administrators can seamlessly transfer or duplicate pages from one application to another. When an administrator chooses to "copy" a page, an identical copy of the page is created in the destination application while the original page in the source application is kept intact. On the other hand, when an administrator selects the "move" option, the page is removed from the source application and is relocated in the destination application.

An administrator can click on the new Move icon within any page section under any particular application to move or copy the page.

From the pop-up window, the administrator can select to which application to move or copy the page.

To copy and not move the page, the administrator can select the **Create a copy** checkbox. If this checkbox is not selected, the page will be moved in its entirety to the target application.

Move Page



Data Set Views

Users have the ability to create Data Set Views (introduced in V11), which will be created based on the Hierarchical Query configuration (dataset and dataset joins).

A Data Set View is an alternate to a data set, that is, any component can be configured using the Data Set View instead of a regular data set. The difference in using a Data Set View instead of a Data Set is that the component will reflect its hierarchical nature; that is, since the self join is inherent to the Data Set View this would impact the resulting component.

An example from Human Resources would be: Let's say we have a Data Set View with an employees and managers relationship. When an individual is searched for, the resulting component will not simply show the records for that individual, as was the case previously with using a Data Set, but will also display all rows for all employees under this individual's organization.

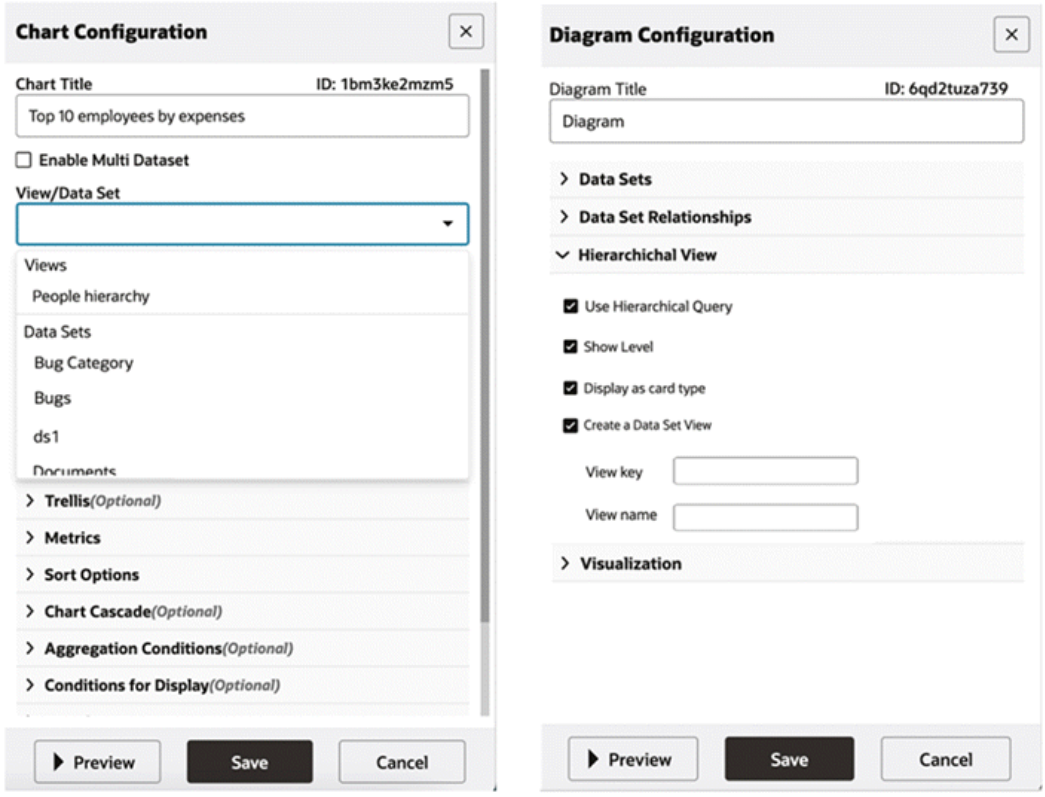
Previously the usage of Hierarchical Query was limited to only Diagram. With this enhanced feature the hierarchical display behaviour is extended to all components.

Aggregation will not be used when creating the hierarchical view. A user can apply aggregations separately on the component later. All attributes from the selected data set will be included.

Only when "Use Hierarchical Query" is enabled will the user see the "Create Data Set View" checkbox along with View key and View Name input options.

Components that use Data Set View should display all rows/items even if they do not contain data. For example, an Aggregate Table should show everyone in the organization even if they do not have any data, unlike they usual behavior.

Examples of Data Set View Options for Chart Configuration and Diagram Configuration

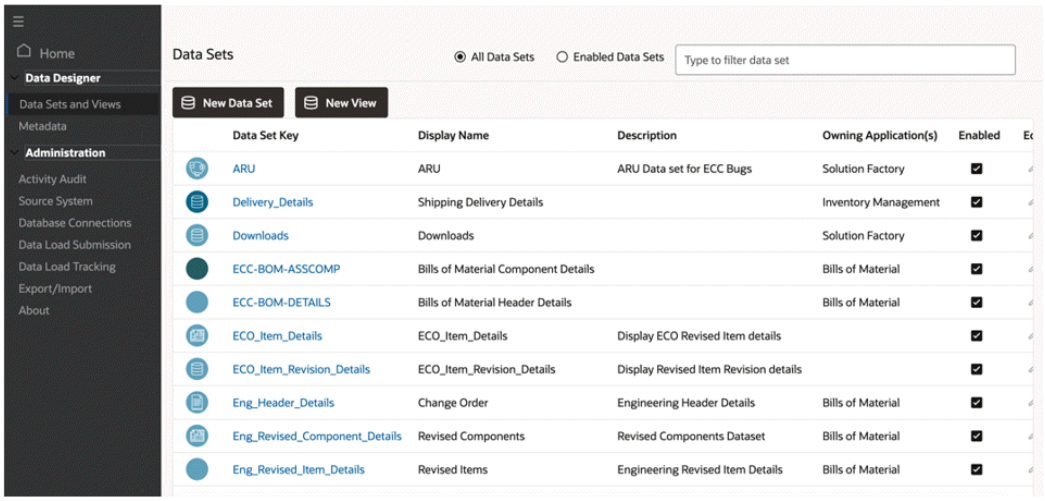


Creating Views in the Administration User Interface

Managing views directly from the administration UI provides for a more standard process to search, create, update, and delete views.

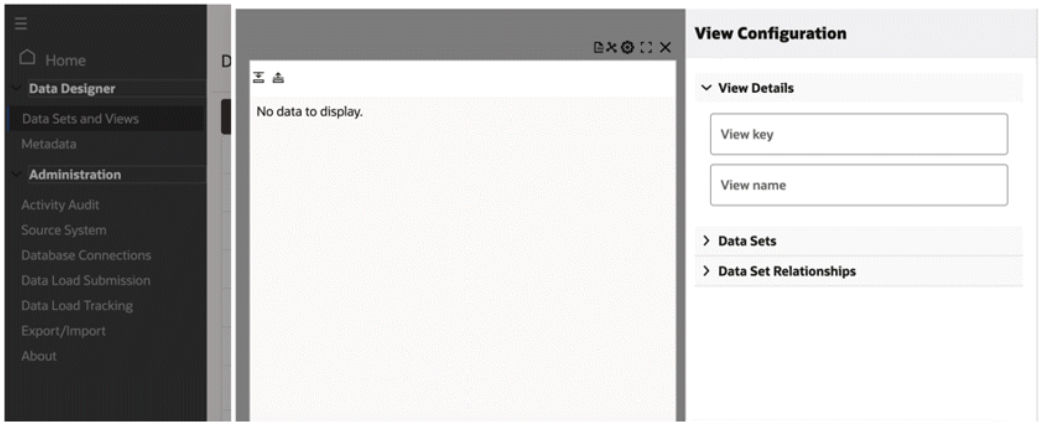
Administrators can create, edit, or delete views from the data set/view accordion. They can also see the **New View** button under the "Data Sets and Views" accordion.

Data Sets Page with "New View" Button for Data Set View



After clicking the **New View** button, the administrator can configure the view within a drawer.

Configuration for a Data Set View within a Drawer



Administrators can configure, preview, and save the view from the within the administration page.

8

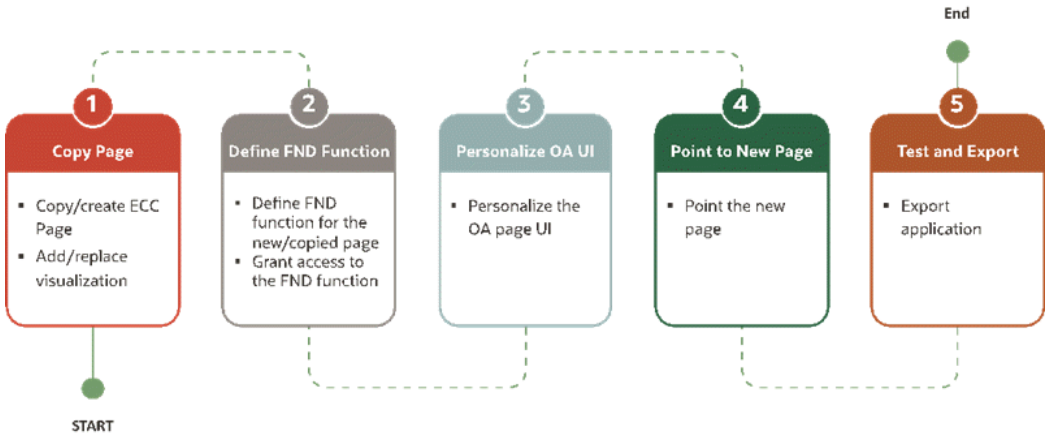
How to Extend Oracle Enterprise Command Centers

Introduction to Extending Oracle Enterprise Command Centers

Because Oracle E-Business Suite is used across many industries and environments, users may have special requirements. In that case, administrative users can extend existing dashboards or create new dashboards based on new data sets.

The following illustration explains the flow of extending an Oracle Enterprise Command Center Framework dashboard:

Flow for Extending Oracle Enterprise Command Center Framework



1. Copy an Oracle Enterprise Command Center page.
2. Define an Oracle Application Object Library (FND) function.
3. Personalize an Oracle Application Framework user interface page.

4. Point to the new page.

5. Test and export the new page.

Basic Extensibility

The goals in this section are:

- Build a new dashboard on an existing data set
- Change the way a component is configured or visualized
- Hide/Show attributes available in the data set
- Remove existing or add new components
- Remove actions or links

Extending an Existing Dashboard:

This section explains how to extend the Oracle Enterprise Command Center Framework dashboard and show it inside the same Oracle Application Framework rich container from a functional administrator perspective.

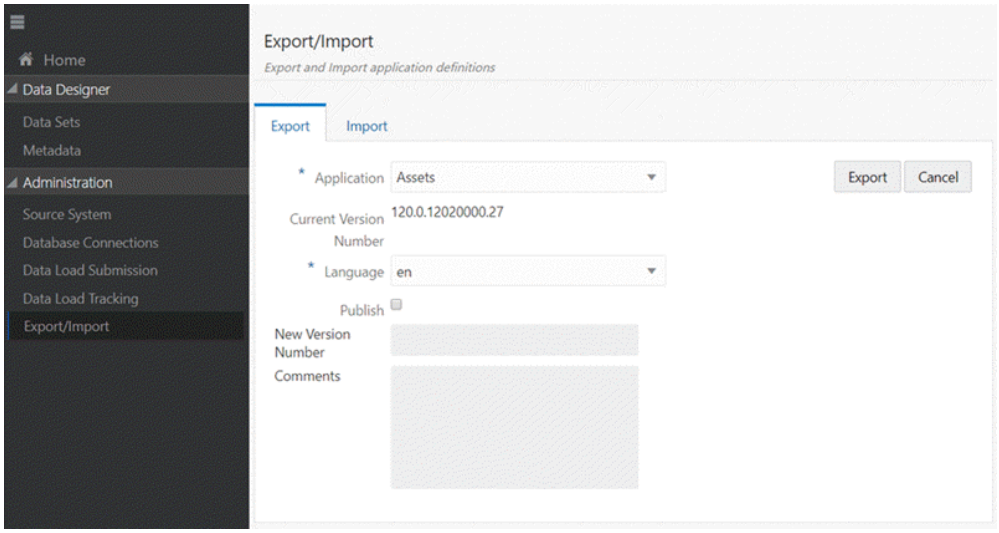
This scenario covers extensions to the Asset Cost dashboard in Assets Command Center as a showcase scenario.

Back Up Existing Application

Before starting extending an application, take a backup from the existing application (shipped) as follows:

1. Using the ECC Developer responsibility, navigate to the Export/Import page.
2. Click the Export tab.
3. Select the application name.
4. Specify the Language: `en`.
5. Click the Export button.

Export Application



Creating a New Page

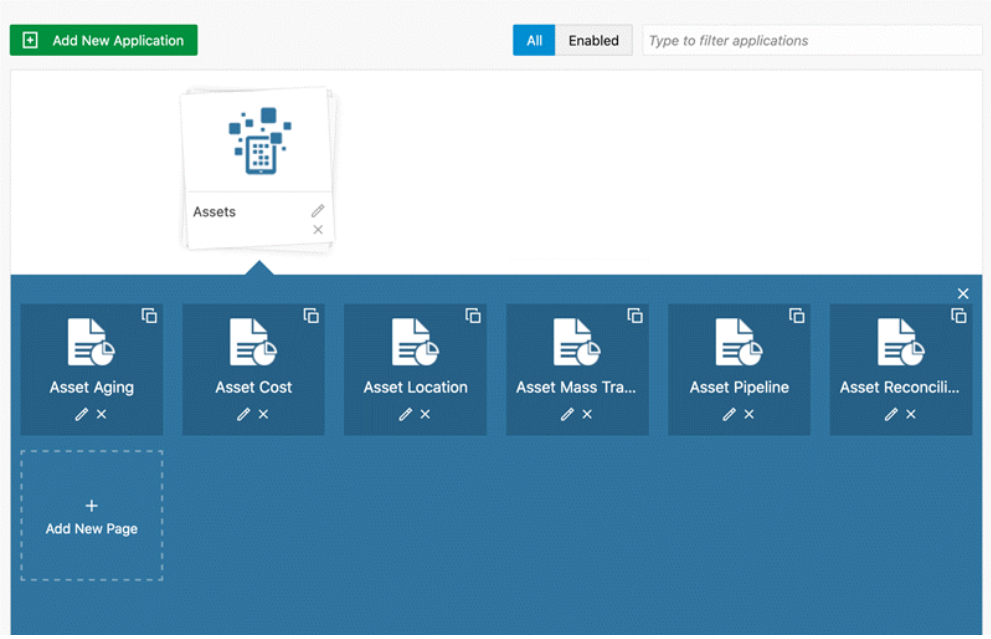
To create a completely new dashboard from scratch, we can either copy an existing page, or completely create a new page to application and then add components in the page.

Copy an Existing Page

As we are targeting to extend the asset cost dashboard, we first need to copy the existing dashboard as follows:

1. Log in with the ECC Developer responsibility,
2. In the Home page, search for the Assets application.
3. Click Assets to explore application pages.

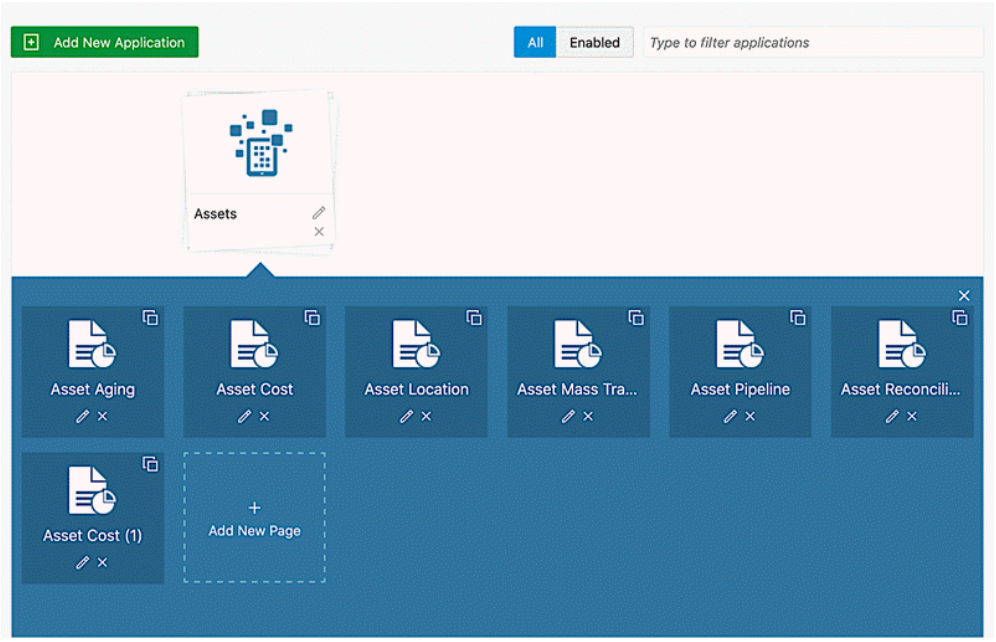
Application Pages



4. Click Copy Page icon in the Asset Cost page.

5. The copied page is added to the application list of pages and named as Asset Cost (1).

New Copied Page

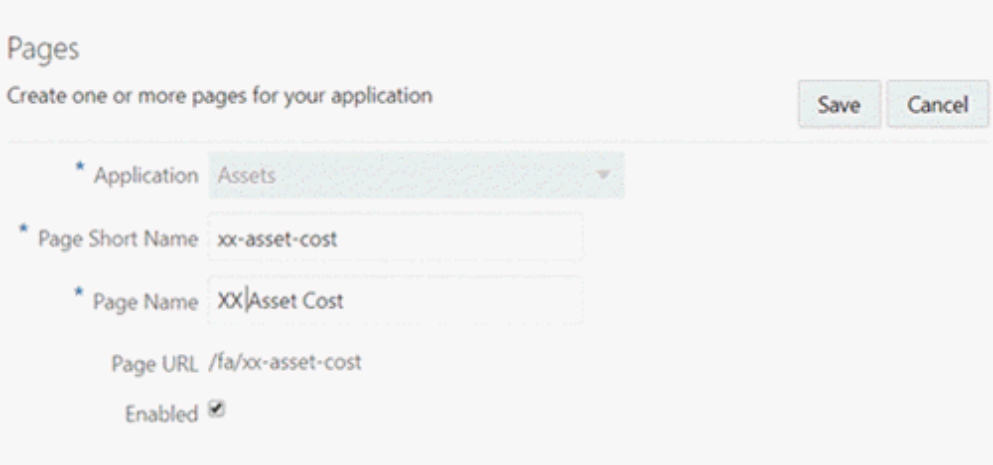


6. Edit the page definition by clicking on the pencil icon on the copied page.

7. In the page definition, change the page key and display name.

   Note: It is recommended to prefix page key and display name with 'xx'.

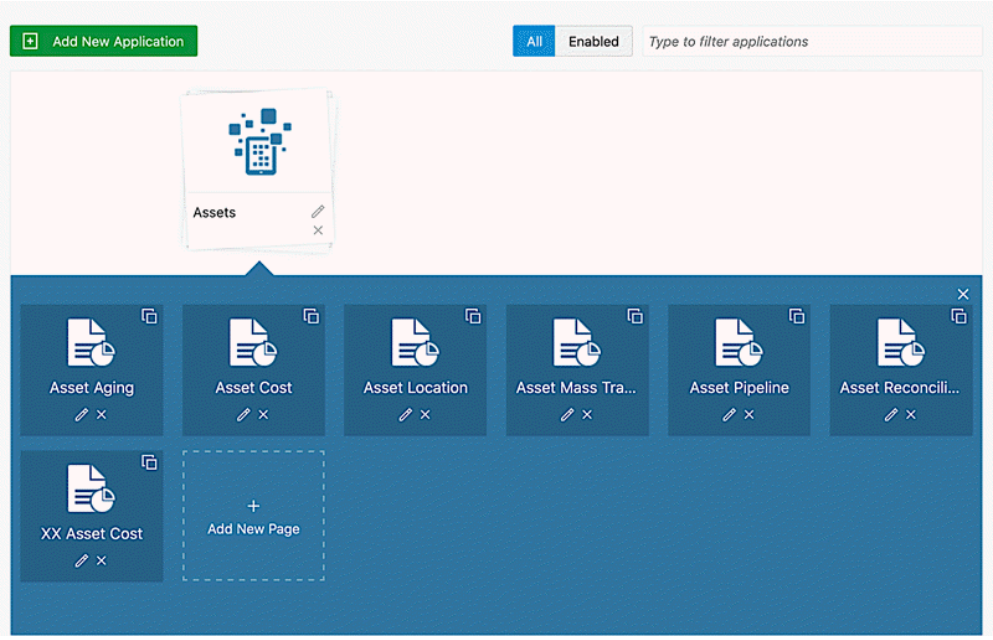
Edit Page Definition



8. Save the changes.

9. Click on the Page name (XX Asset Cost), open the page, and start the extension.

New Copied Page



Create a New Page

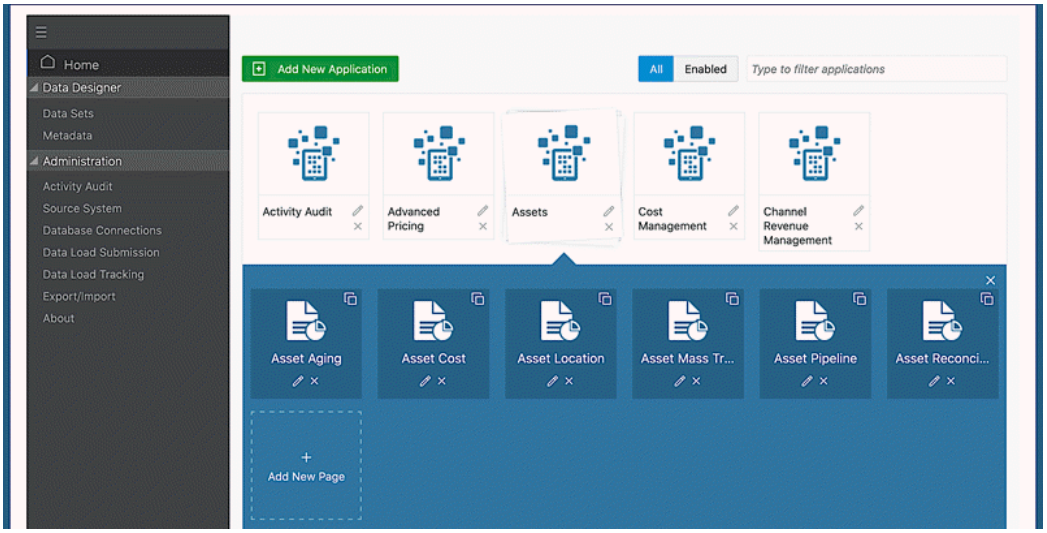
1. Using the ECC Developer responsibility, navigate to the ECC home page.

2. Click on an Application

3. Click on "+ Add New Page" to create a blank page.

Add a New Page

ECC Administration Page with '+Add New Page' Icon



Page Creation Guidelines

- Selected Refinements to be added in the dashboard mandatorily.
- Place the search box inside the side navigation panel or at the page level.
- The Available Refinements component is added by default in the default page layout or collapsed side navigation layout.
- If the page layout is "No side nav", then the Available Refinements component will not be added in the page.
- You can add only one search component and selected refinements.
- You can use a tab component to organize the visualization components in the page.
- You cannot add a tab component inside another tab component (nested tabs are not allowed).

1. In the Page, enter the page short name, and the page display name.
2. The page URL is generated and displayed by default in "Page URL:".

3. Choose the page layout:
   - Default Side Nav: Side navigation listing the refinements is always visible when user accesses the dashboard.
   - Collapsed Side Nav: Side navigation on page is collapsed.
   - No Side Nav: Side navigation is not available. This is suitable for pages that do not require additional user refinement.
4. Click on Save to create the new page.

Extend a Page

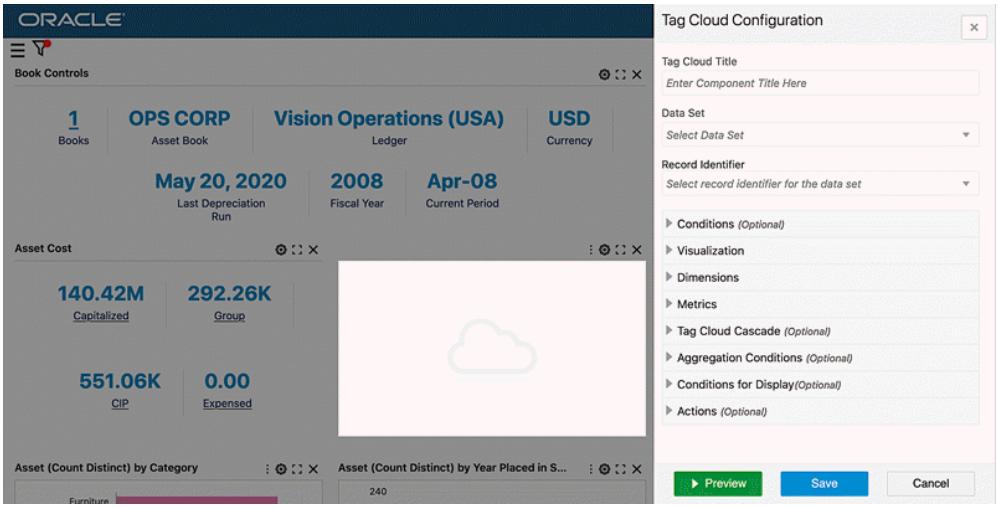
Based on the functional requirements to enrich the Asset Cost dashboard, the extension targets the following scenario:

Replace Existing Visualization

Replace Asset Cost summarization bar with a tag cloud, where the user has asset type term is displayed based on the asset cost and cascade down to property type.

1. Open Add Component List. Drag and drop the tag cloud component beside the Asset Cost summarization bar.
2. Click the Configuration icon.

Tag Cloud Configuration



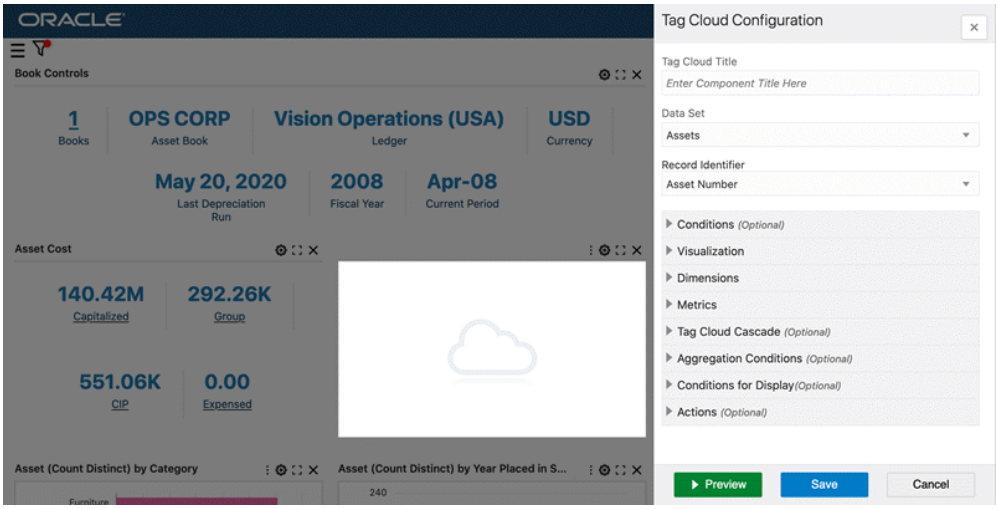
3. Configure Tag Cloud:

Tag Cloud Configuration Options

Option	Description
Title	Leave the title empty, to display dynamic title based on dimension and metric are defined.
Data Set	Assets
Record Identifier	Asset Number **Note:** The record identifier prevents aggregate duplicate records based on the granularity level in the data set.
Condition	NA
Visualization	Enable Show Metric Value Number of Items = 10
Dimension	Asset Type Property Type
Metric	Asset Cost (Sum) Asset (Count Distinct)
Tag Cloud Cascade	Asset Type Property Type
Aggregation Condition	NA
Condition for Display	Book Type Code (Count Distinct) = 1 **Note:** As the tag cloud will display the asset types based on cost, then it is better to display the asset cost within the context of asset book.

4. Click Preview to view the component runtime view.

Tag Cloud Configuration Preview



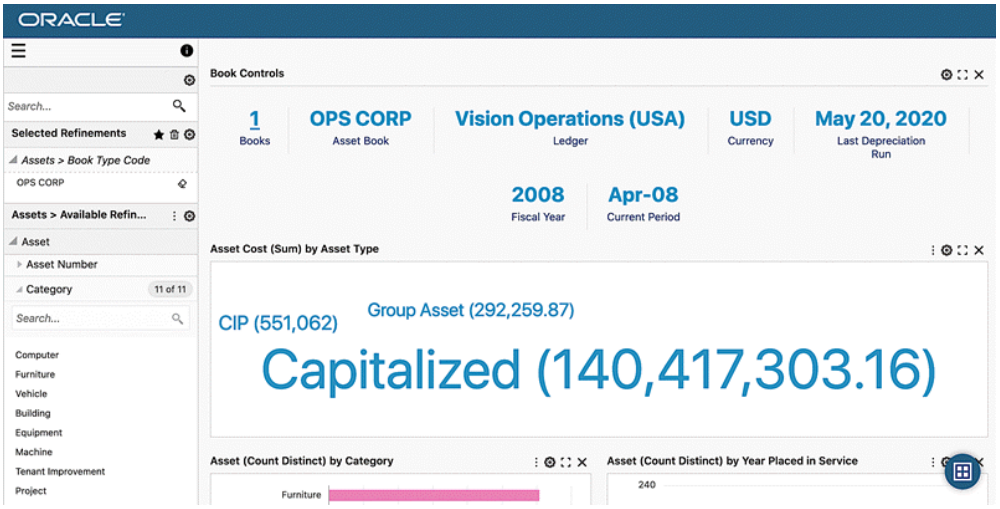
Note: As there is no asset book selected yet, the tag cloud displays "No data to Display" until the user selects one asset book.

5. Click Save. Select Asset Book from Summarization Bar "Flag pop-up" to get the data displayed within the context of the selected Asset Book.

6. Delete Asset Cost summarization bar.

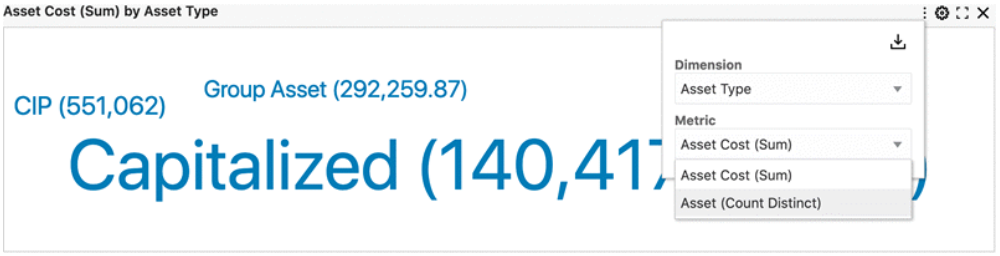
   Click on the Delete icon.

Asset Cost (Sum) by Asset Type Tag Cloud

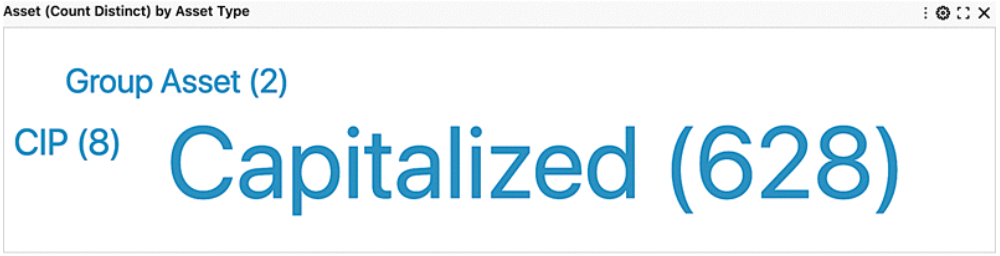


7. Using the runtime option, the user can change the metric displayed on the tag cloud. Click Option, and then change the metric value using the drop-down list.

Drop-down List to Change Metric Value



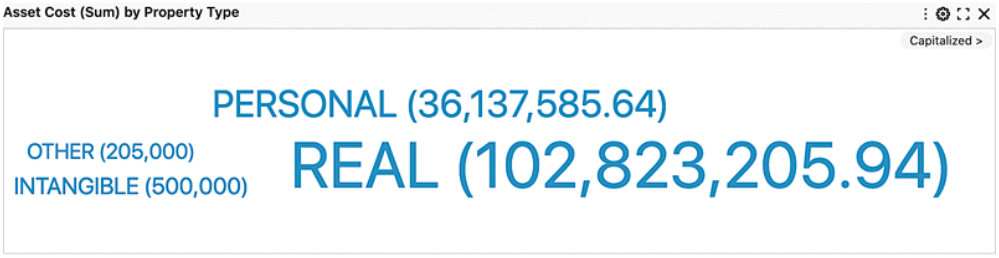
Asset (Count Distinct) by Asset Type



8. When a user filters by any term in the tag cloud, the filter will cascade down to the Property type code.

   Click to filter.

Tag Cloud Cascading Filter



Update a Chart

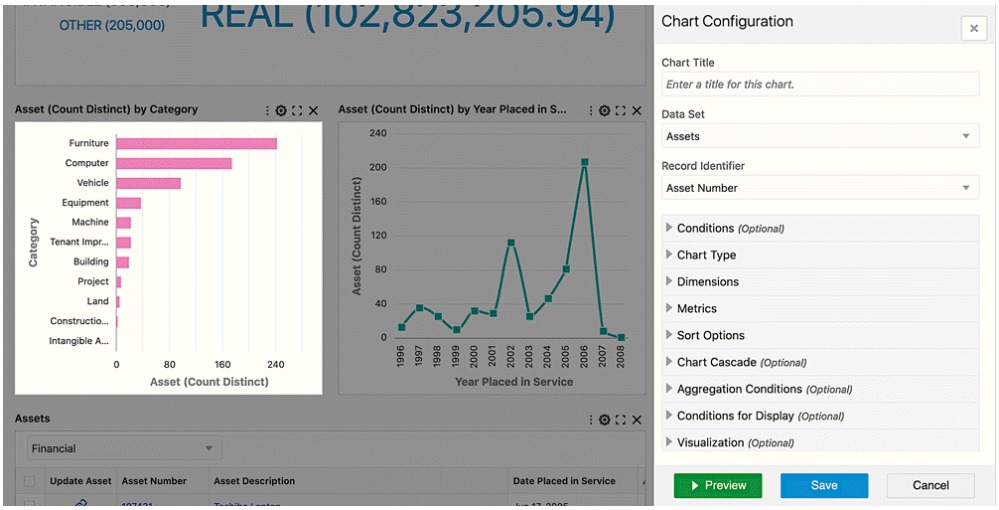
Adding a reference line at the average total cost by category so that users can compare total asset cost per category against the average.

In this extension, the following steps will be done:

- Add a reference line.
- Change chart configuration to display asset cost first instead of count.
- Define a new condition for display based on Asset Book.

1. Click Configuration to open the chart configuration panel.

Edit Chart Configuration



2. Open the Metric configuration option, and update the following:
    1. Delete the first metric defined, Asset (Count Distinct).

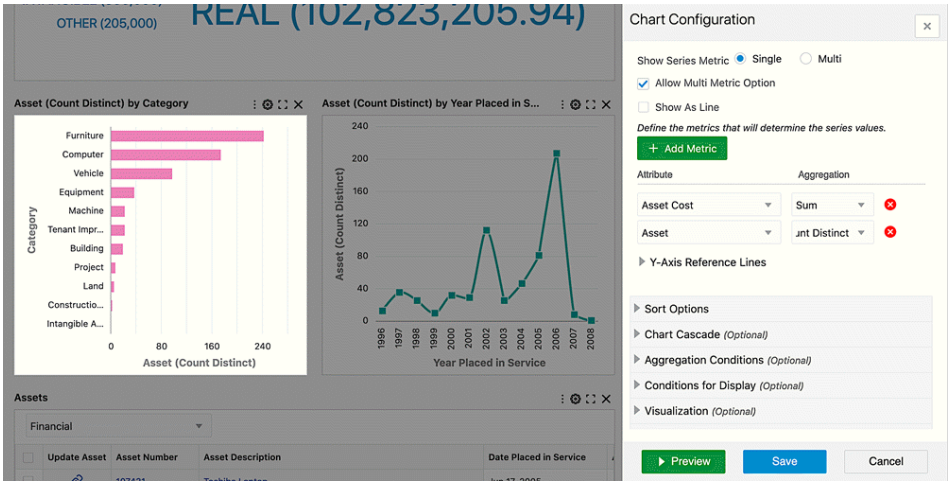
2. Click + Add Metric.

3. Define Asset (Count Distinct).

4. Click Preview.

   The chart displays Asset Cost (Sum) at runtime.

Updated Metric Configuration



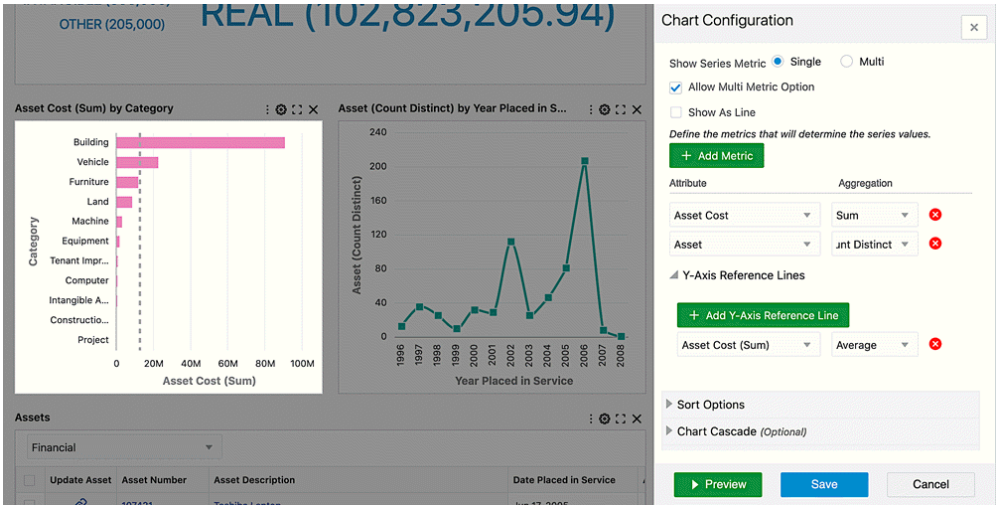
3. Click +Add Reference Line.

   1. Select the first metric defined "Asset Cost (Sum)".

   2. Select Aggregation function "Average".

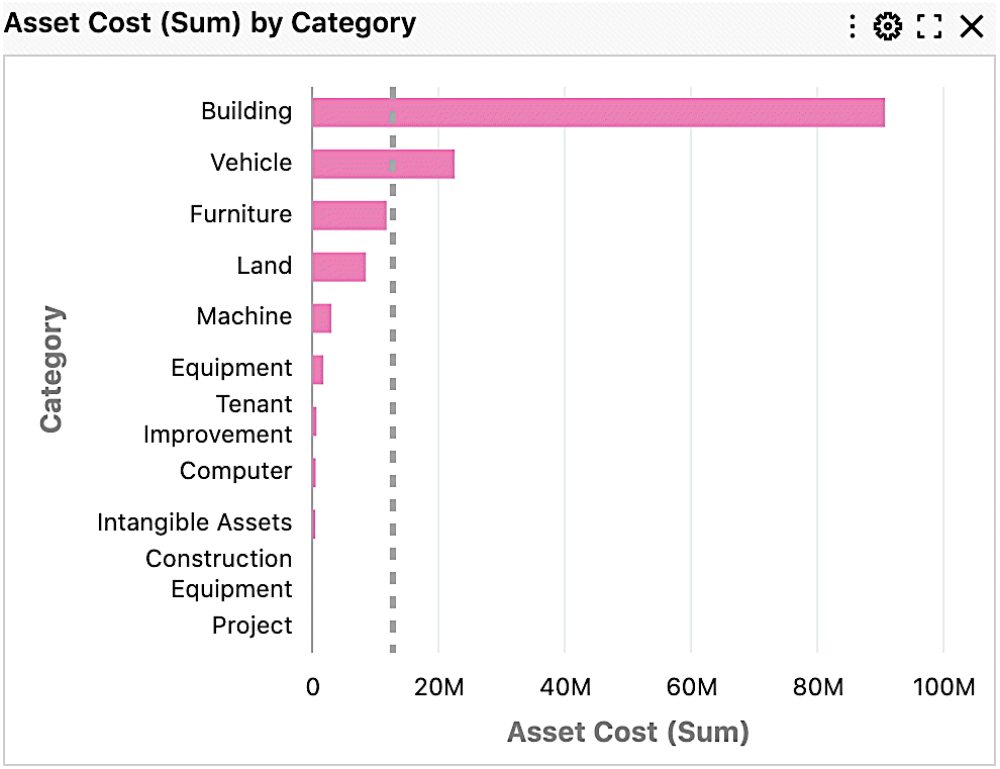
   3. Click Preview, and the reference line based on average total cost by category is displayed on the chart.

Reference Line Configuration



4. Save the Chart Configuration.

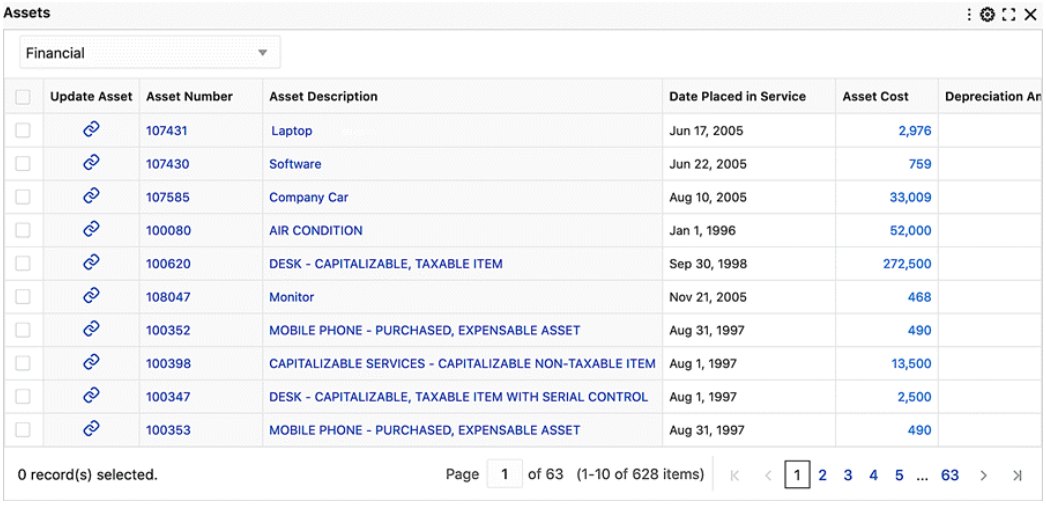
Reference Line



Update a Results Table

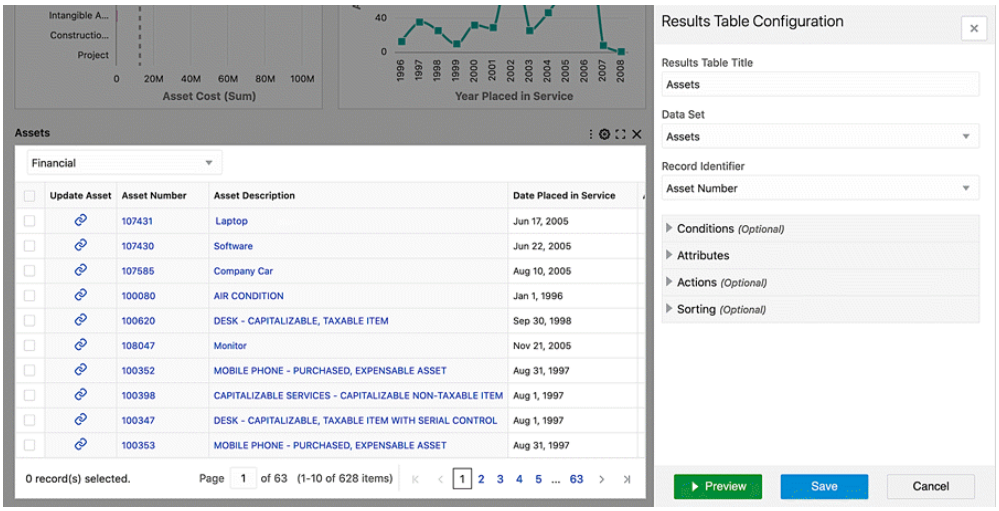
Define a Hyperlink Action in Asset Cost dashboard to navigate to Asset Location dashboard.

Assets Results Table



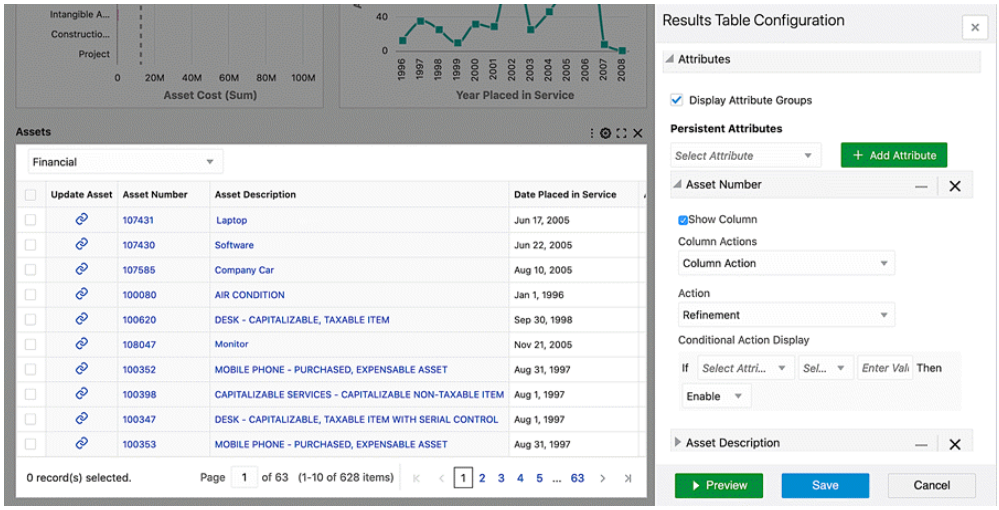
1. Click the configuration icon to open the configuration panel.

Results Table Configuration



2. Expand Attribute according to define the action at the attribute level.

Column Action Configuration



3. Apply the following configuration information:

Column Action Configuration Information

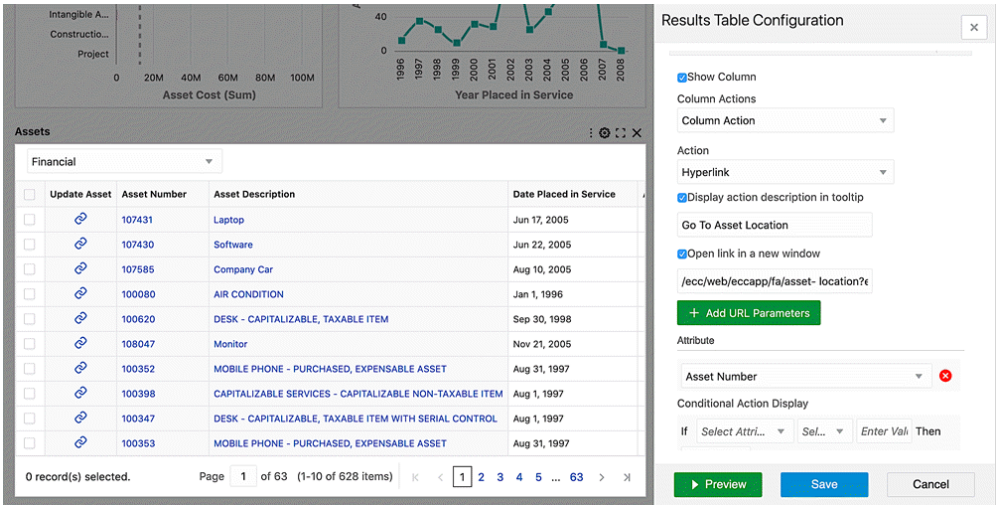
Option	Value
Column Action	Column Action
Action	Hyperlink
Display action description in the tooltip	True In the tooltip test box: Go to Asset Location
Open in a new window	False
URL*	`/ecc/web/eccapp/fa/asset-location?eccDeepLink=[{"fa-asset":[{"attributeKey":"ASSET_NUMBER","values":[{0}],"operator":"=="}]}]`
+ Add URL Parameter	Select attribute: Asset Number Note: asset number reference {0}, is added in the URL

* Deep link:

- Used to navigate between two different pages and apply a filter(s) to the destination page
- Can apply a range filter, selection filter, and search filter

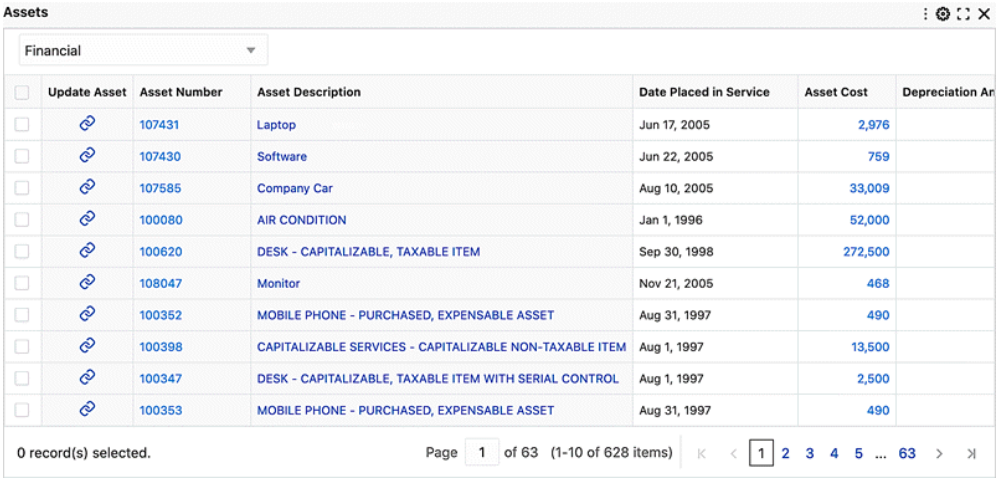
4. Click Preview and Save.

Previewing and Saving Configuration

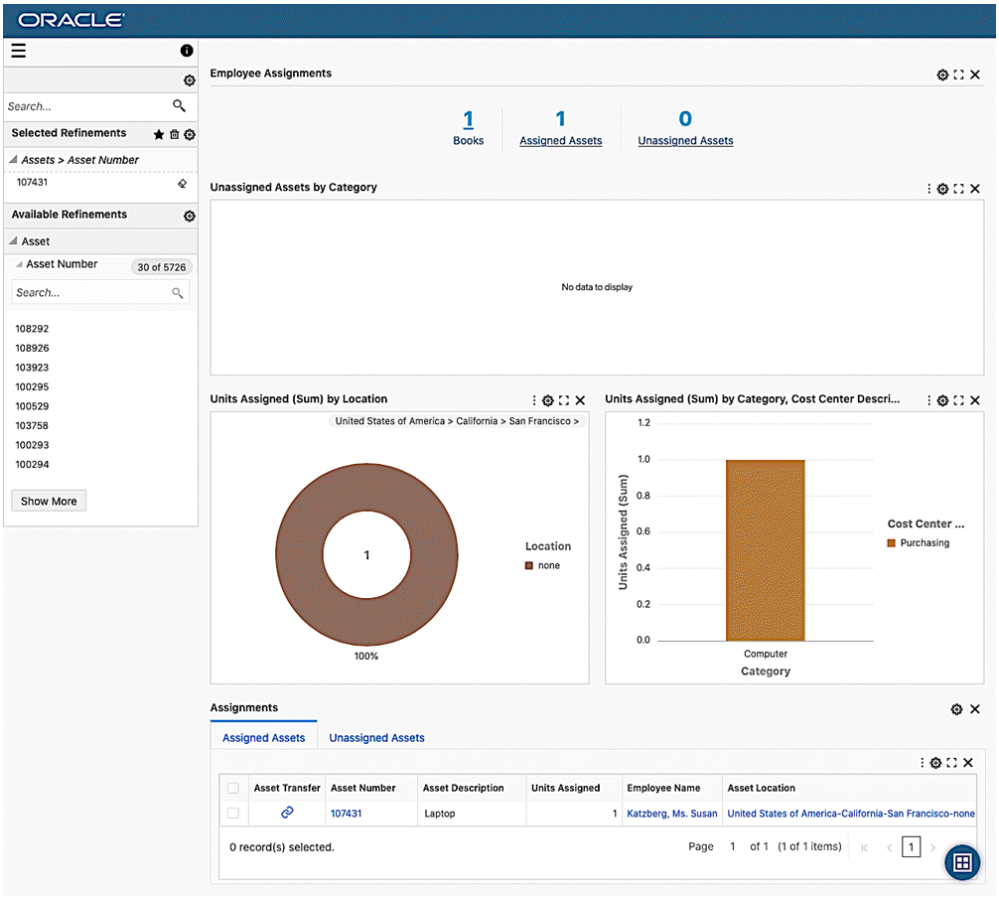


5. At runtime, a user can click an asset number to navigate to the asset location dashboard, and then filter the dashboard with the selected asset number.

Example Asset Location Dashboard



Example of Filtering on Asset Number



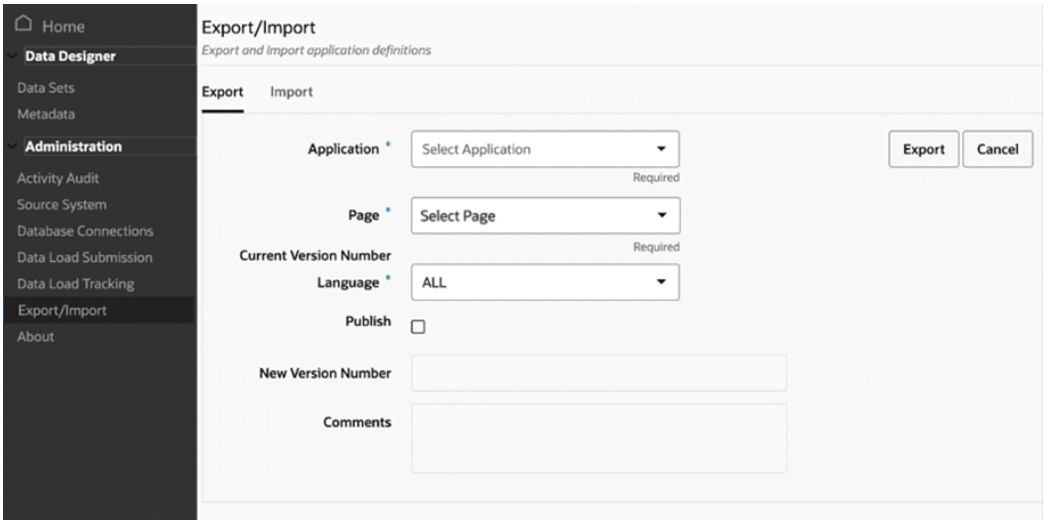
Collaboration Activities

Collaboration features include exporting an application, and copying/moving pages across applications.

Exporting an Application

Administrators can export a subsection of pages within an application in Oracle Enterprise Command Center Framework.

Export/Import Page



Copying and Moving Pages across Applications

Administrators can also copy or move pages across applications in Oracle Enterprise Command Center Framework.

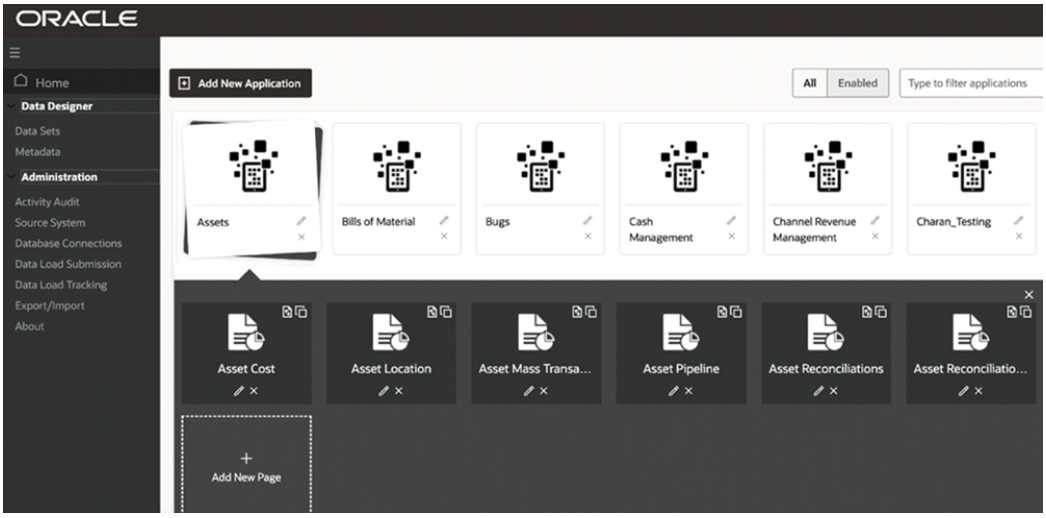
When copying a page, the page will be copied to the destination application while still being available in the source application.

When moving a page, the page will be moved to the destination application and the source application will no longer have that page.

If an administrator moves a page from one application to another, then for them to access the data set from this application, it should be enabled at the security handler level; that is, with the page function to access the data set.

This functionality enables administrators to move pages across applications which means developers can reuse existing pages. This feature thus reduces development time and effort.

Export/Import Page



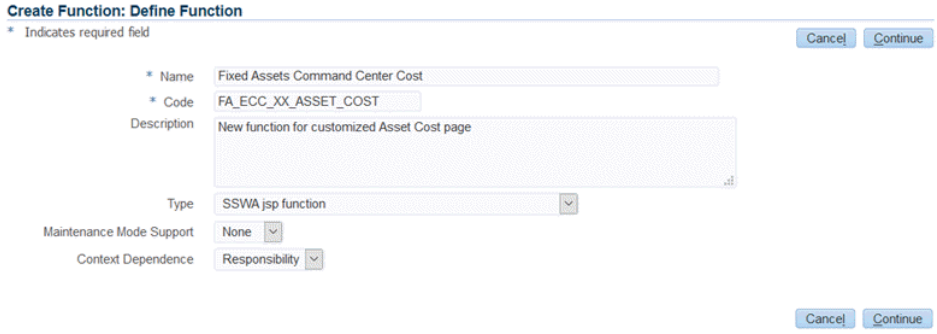
Personalizing in Oracle E-Business Suite

Create FND Function:

After applying all the required extensions in the copied Asset Cost dashboard, create a new FND function and grant access to this function as follows:

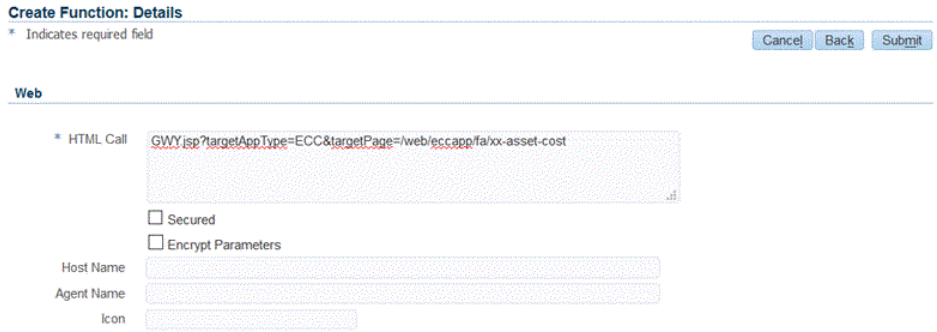
1. In the Oracle E-Business Suite environment, navigate to the Functional Administrator responsibility.
2. In the Core Services tab, select the Function subtab.
3. Select the Create Function tab.
    1. Enter the function code, 'FA_ECC_XX_ASSET_COST'.
    2. Enter the function name, 'Fixed Assets Command Center Cost'.
    3. For Type, select 'SSWA jsp function'.
    4. Click the Continue button.

Creating a Function



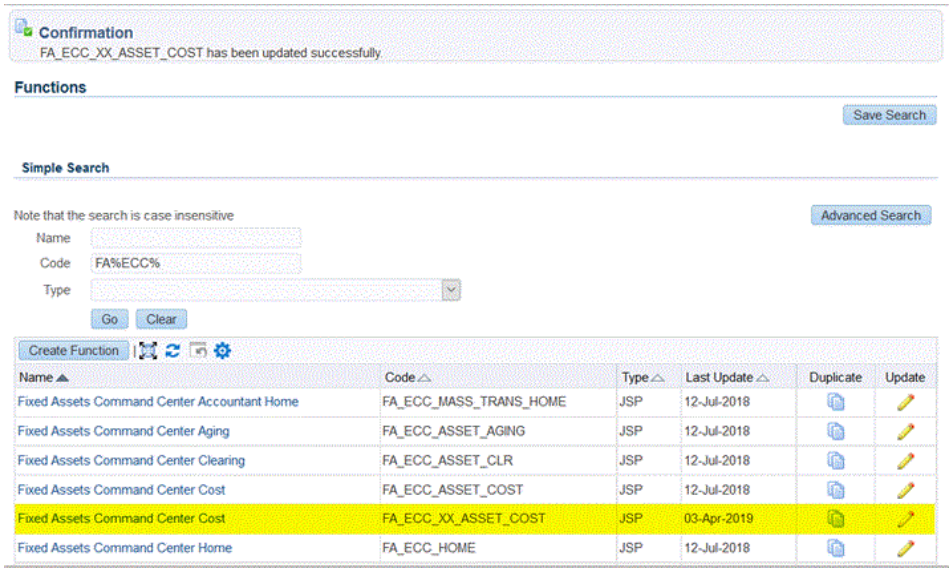
5. Enter the page URL in HTML Call text box. Note: in the Page URL, enter the page key created in the section Copy Existing Page, page 8-3..

Example of a Page URL for a Function Definition



6. Click the Submit button.

Confirmation of Function Creation



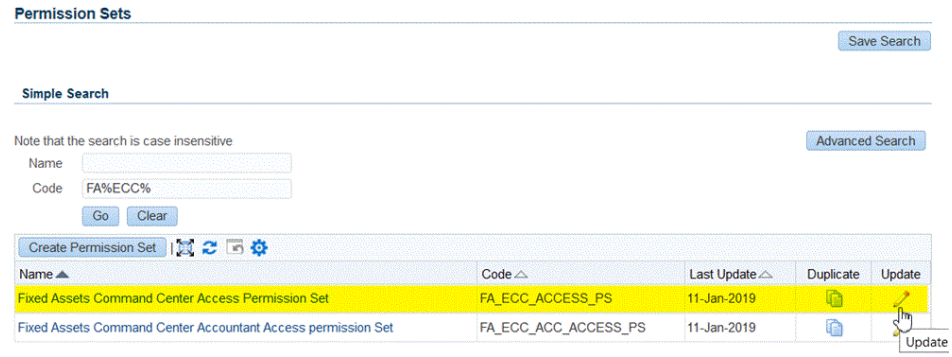
4. Navigate to the Security tab and select Permission Set.

5. Assign the new FND function to the Assets Permission Set.

    1. Search for permission set "Fixed Assets Command Center Access Permission Set" - "FA_ECC_ACCESS_PS".

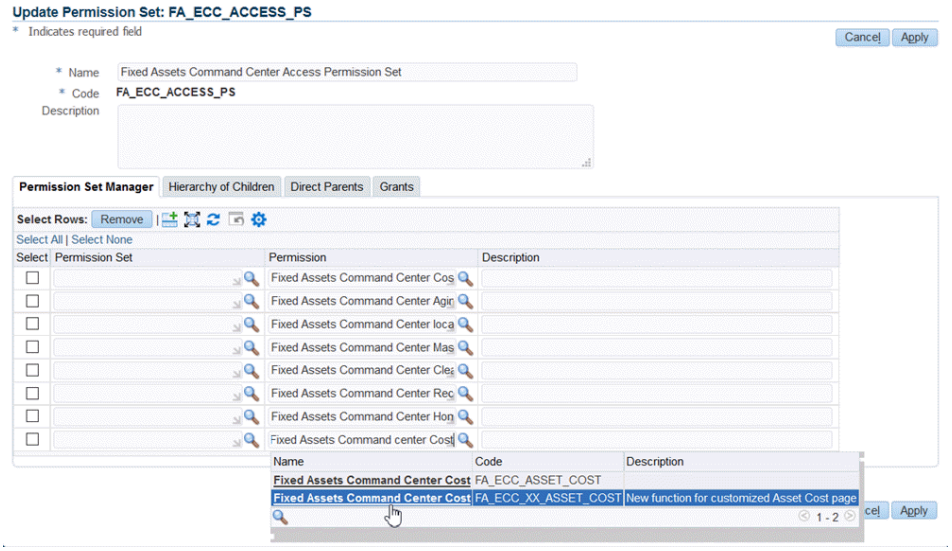
        Click Edit.

Example of a Permission Set



    2. Add the newly created function.

Updating a Permission Set



3. Click the Apply button.

Personalize the Oracle Application Framework Page:

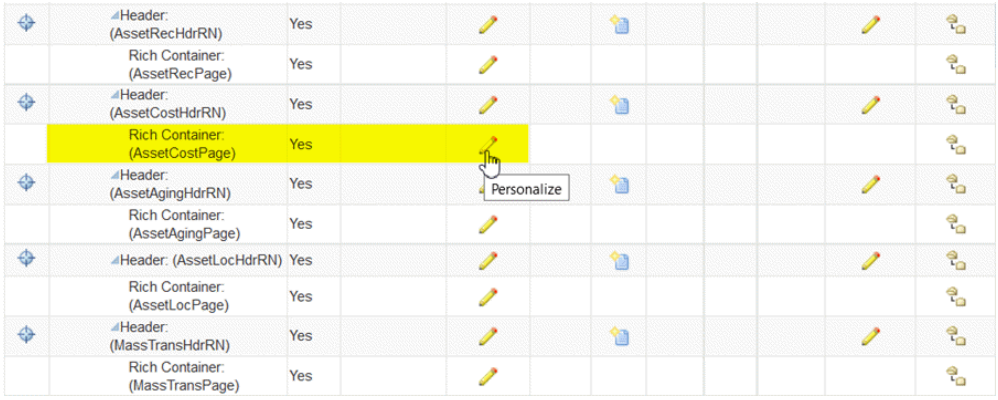
Now, you need to personalize the Oracle Application Framework page to replace the shipped dashboard with the new personalized one, using the steps below.

1. In Oracle E-Business Suite, navigate to the Assets Command Center, then to the Asset Cost page (shipped page).

2. At Oracle Application Framework page level, choose **Settings**, then **Personalize Page**.

3. Click **Edit** to personalize the Asset Cost page rich container.

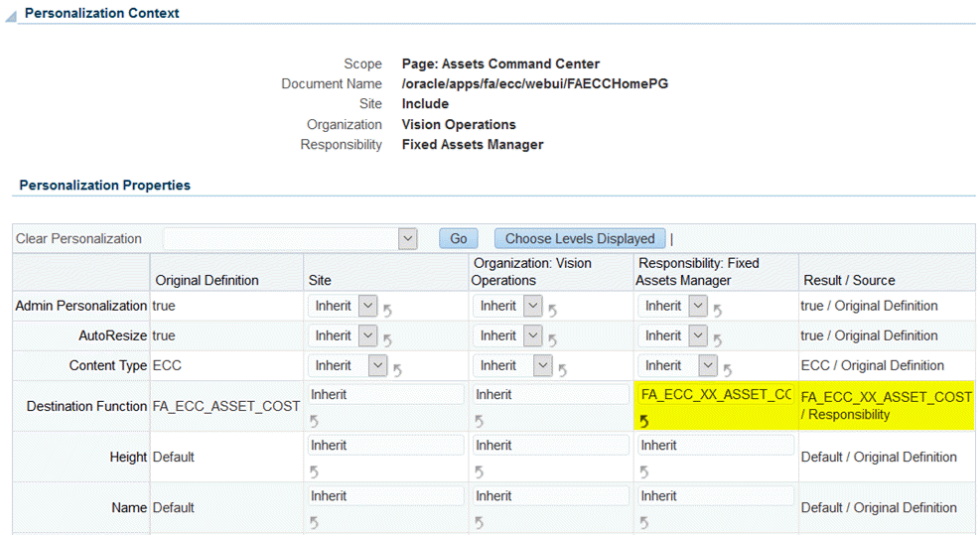
Personalizing the Asset Cost Page Rich Container



4. At the responsibility level, personalize the rich container by adding the newly created function FA_ECC_XX_ASSET_COST in the Destination Function.

   Note that you can personalize the page at the site or organization level as well.

Personalizing a Page at the Responsibility Level



5. Click the **Apply** button.

Test and Export:

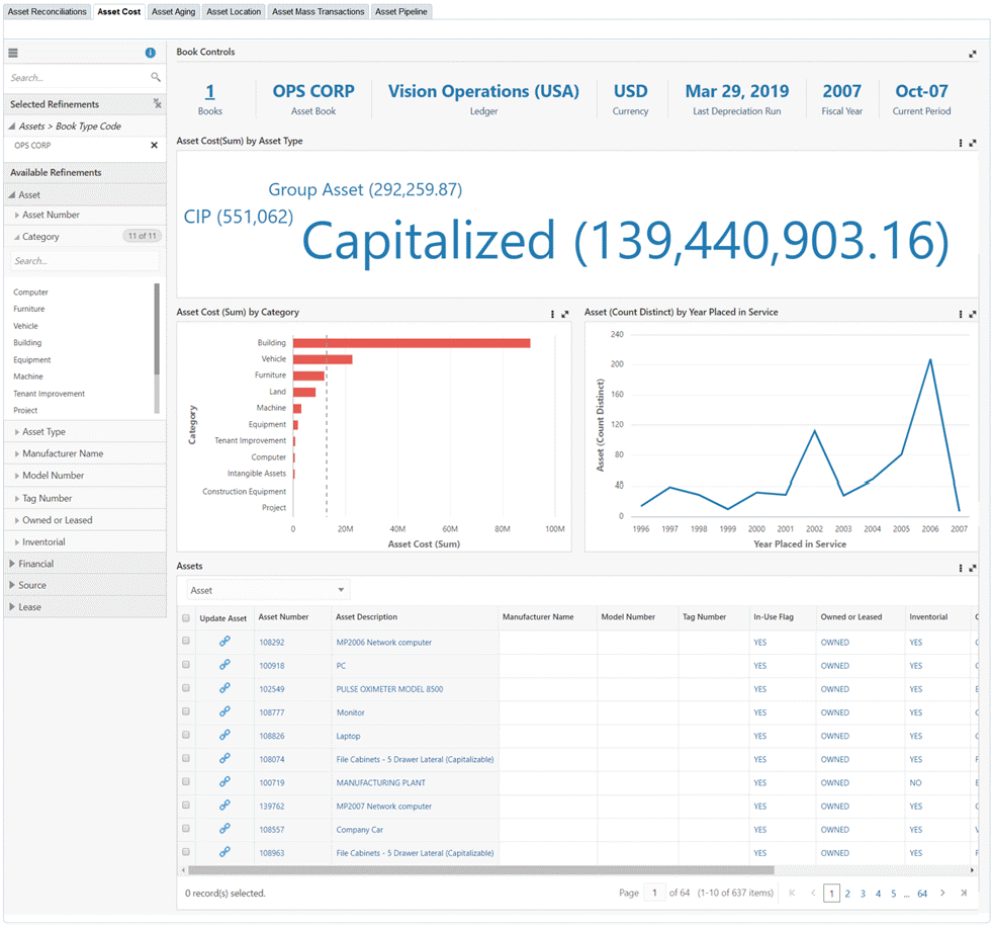
After personalizing the Oracle Application Framework page, access the new extended page from Oracle E-Business Suite.

1. Navigate to the Fixed Assets Manager responsibility.

2. Navigate to the Assets Command Center and select the tab for the Asset Cost dashboard.

3. The newly extended dashboard is displayed instead of the shipped Asset Cost dashboard.

New Asset Cost List

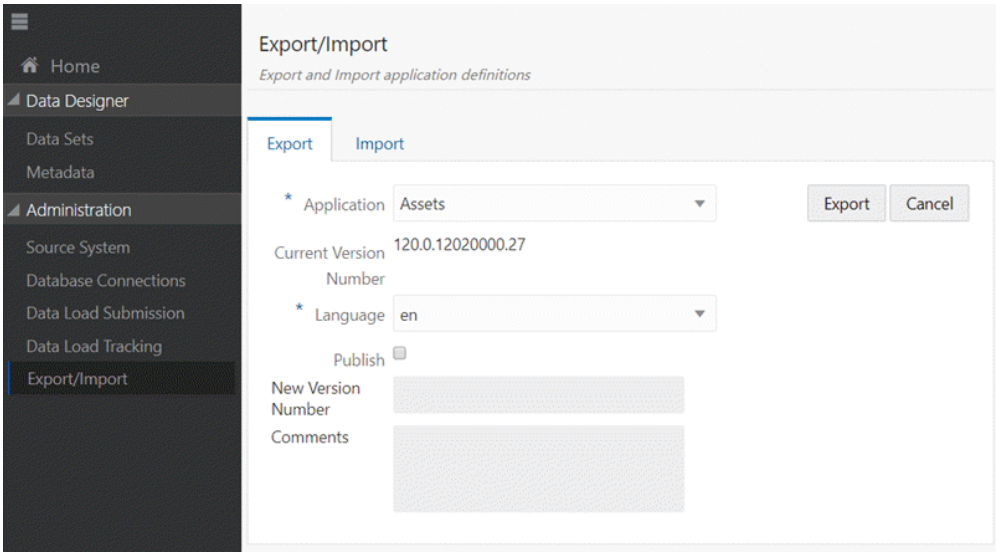


4. Validate the dashboard functionality.

5. Navigate to the ECC Developer responsibility.

6. Navigate to the Export/Import page and click the Export tab to export the updated application.

Exporting the Updated Application



Adding a New Tab to a Command Center Oracle Application Framework Page

To add the extended dashboard as a new tab for a command center, follow one of the methods below, based on the command center.

Add a New Tab in an Existing Oracle Application Framework Page:

To add a new tab to the existing Oracle Application Framework Page, use the personalization capabilities supported in Oracle Application Framework in Oracle E-Business Suite. Refer to Personalizing Your Pages and Portlets, *Oracle Application Framework Personalization Guide* for more information.

Prerequisites for this procedure are:

- Personalization must be enabled in accordance with the steps in the *Oracle Application Framework Personalization Guide*.
- The Oracle Application Framework Page must have `subTabLayout` already configured.
- An Administrator has created a new function (using the Functional Administrator responsibility) pointing to the custom ECC Dashboard page.
- The newly-created function is added to the permission set of the respective command center access role. Otherwise, the end user is not authorized to view the dashboard page.

The following steps provide an example of adding a new tab to the existing Assets Command Center.

1. Log in to Oracle E-Business Suite as a user with the Functional Administrator responsibility.

2. Navigate to the Functional Administrator responsibility, then Personalization.

3. Select the Personalization tab to display the Application Catalog.

4. Specify the Application and optionally, the Document Path.

5. Select the **Edit** (pen) icon in the Personalize Page column for the page you want to personalize.

6. Choose the personalization context, then click **Apply**.

7. Navigate to Personalize page.

    1. The Personalization Context shown at the top of the page lists the context for the top-level object, which in the case of the Oracle E-Business Suite, is the page layout. It identifies the scope of the personalization you are about to make, as well as to which personalization level(s) and value(s) these personalization apply.

    2. The Personalization Structure is shown the Page Hierarchy Personalization page in the simple view. Note that the children of a node do not reflect the actual hierarchy structure of the children (because some layout elements are not shown); therefore, the **Reorder** and **Create** icons are disabled.

8. Switch Personalization Structure to Complete View to enable the **Reorder** and **Create** icons for the children of that node, so that the correct hierarchy of the children is shown in the Complete view.

9. Create a new page layout region on the Command Center Oracle Application Framework page.

10. In the new page layout region, create a `subTabLayout` region.

11. Launch the **Create Item** page where you can create a new child item for this region.

12. Create an Item at the Site level with an Item Style of Header. Enter a header ID, for example, `XX_ASSET_COST`. Click **Apply**.

13. In the Header region, create a subtab by clicking the **Create Item** icon.

14. Launch the **Create Item** page to create a new child item in this region, of style "Rich Container".

Enter the following:

- Provide an Item ID. For example, `ECC_XX_ASSET_COST`.
- Choose **ECC** for the Content Type.
- For Destination Function, specify the function name of the ECC dashboard page. For example, `FA_ECC_XX_ASSET_COST`.
- For Title, enter a title. For example, `XX Asset Cost`.

When finished, click **Apply**.

15. Create a link by clicking on **Create Item** at the "Sub Tab Bar: (subTabBar1)" level.
16. Launch the **Create Item** page to create a new child item in this region, of style "Link."

    Enter the following:

    - Item ID
    - Destination function

    Then click **Apply**.
17. The new tab is now created for the new custom dashboard.
18. The dashboard can be accessed from the functional responsibility

Adding a Dashboard as a new Oracle Application Framework Page:

1. Add a new submenu to the existing command center menu.
2. Configure the function for the extended dashboard on the new submenu (the function should point to the dashboard Oracle Application Framework page).

Advanced Extensibility

Extend an Existing Data Set

Objectives covered in this section are:

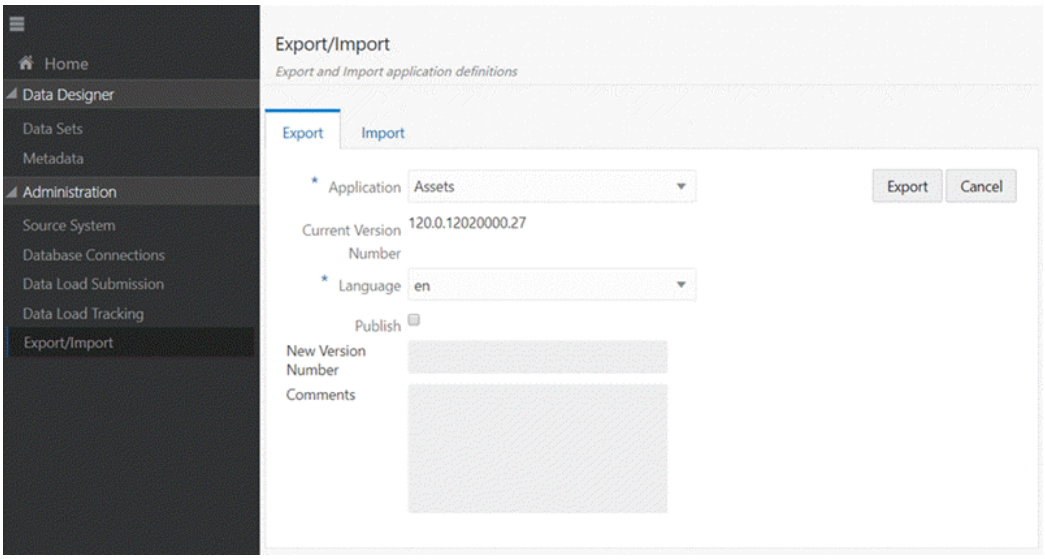
- Extend a data set to include additional attributes not shipped out of the box but available in Oracle E-Business Suite.
- Display those attributes in the different part of the Oracle Enterprise Command Center Framework dashboard.

- Add additional attributes
- Back up an existing application
- Create a new data set

Before starting extending an application, take a backup from the existing application (shipped) as follows:

1. Using the ECC Developer responsibility, navigate to the **Export/Import** page.
2. Click the **Export** tab.
3. Select the application name.
4. Specify the Language: `en`
5. Click the **Export** button.

Export an Application



Shipped data set can be extended by creating additional attributes from source data.

Perform the following steps to include new attributes. (For examples on working with metadata load and data load procedures, refer toCode Example for Designing a Metadata Load Procedure, page D-2 and Code Example for Modifying a Data Load Procedure, page D-4.

1. Review the ECC_SPEC_ID structure for the shipped data set. Inclusion of new attributes should not change the grain level of data. If new attributes affect the grain level, then the ECC_SPEC_ID structure has to be revised.

2. Copy the PL/SQL procedure in the data load for the shipped data set.

3. If the data load procedure uses a database view, copy the view.

4. Update the query in the data load procedure in order to ingest the data into the new attributes.

5. Update view name in PL/SQL procedure for both the full load and the incremental load.

6. To create new attributes in metadata:

   - If the data set has a shipped metadata load, copy the shipped PL/SQL procedure and update the procedure to include the new attributes.

   - If the data set has no shipped metadata load, create a new PL/SQL package and procedure to create new attributes.

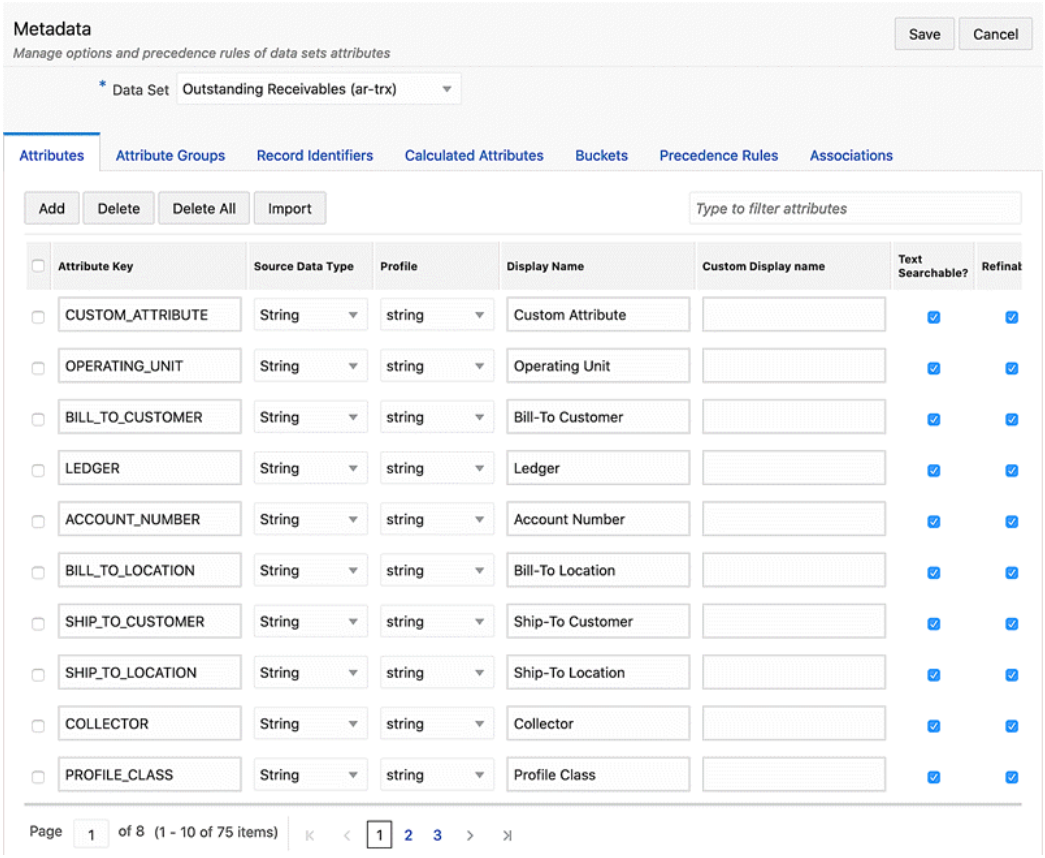
7. Navigate to the Data Sets section in the Administrator UI, open the data set and add the new metadata load and data load.

   Note: The sequence for new load rules should start with 101, as sequence numbers till 99 are reserved for shipped data loads.

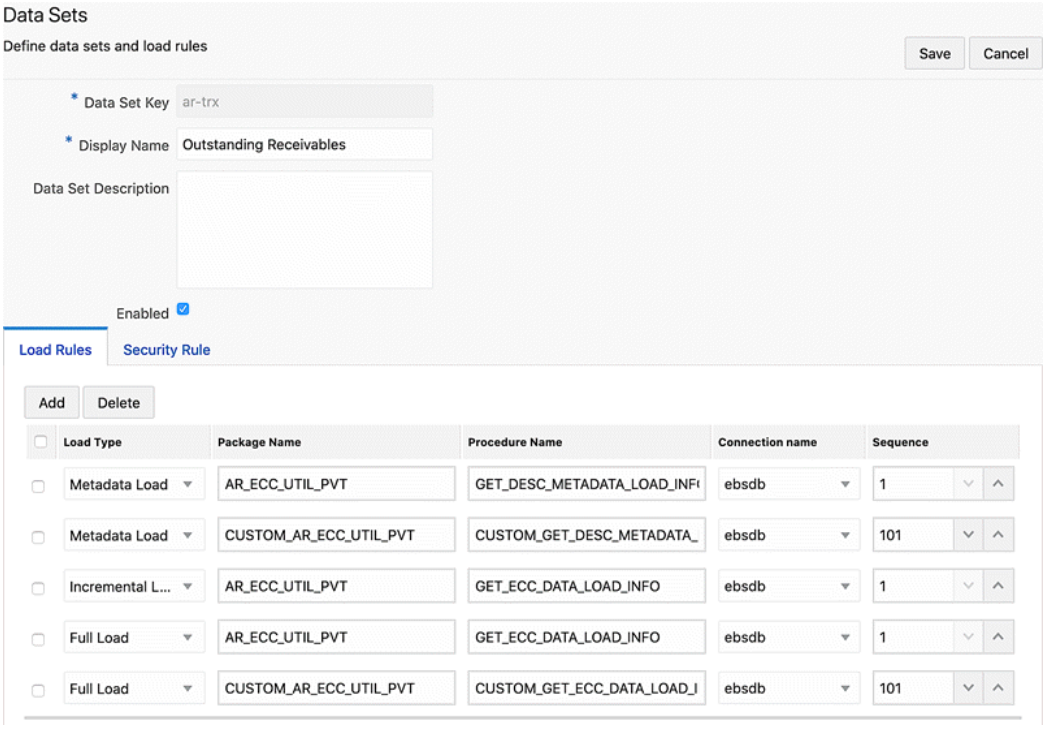
8. For validating the new load rules, run the metadata load followed by a full load.

9. Once the loads are successful, update the data load concurrent program by extending the executable file of concurrent program.

Example of New Attributes in Metadata



Example of Updated Data Set Load Rules

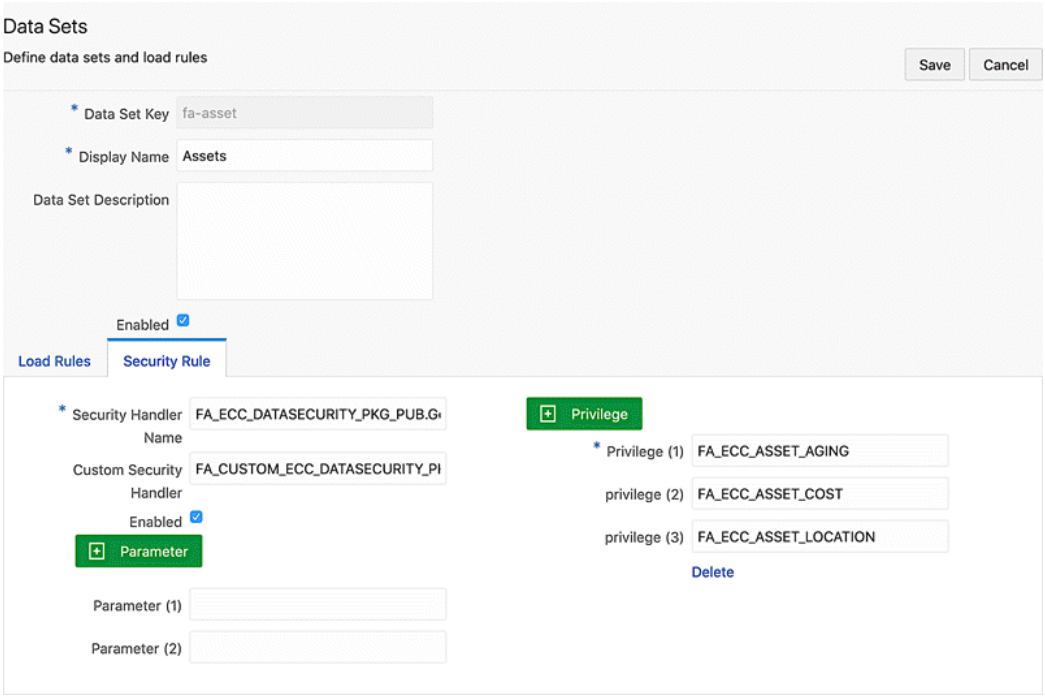


Add a Custom Security Handler

You can extend a shipped security handler by including a custom security package in the data set security rule. The filters from the custom security handler are combined with filters from the shipped security handler. Perform the following steps to include a custom security handler. (For details on sample metadata load and data load procedures, refer to Code Example for Designing a Metadata Load Procedure, page D-2 and Code Example for Modifying a Data Load Procedure, page D-4.

1. Copy the shipped security handler PL/SQL package and procedure.
2. If the procedure uses a database view, copy the view.
3. Update the query in security handler procedure.
4. Update the view name in PL/SQL procedure. Save the package and procedure.
5. Navigate to the data sets section in the Administrator UI, open the data set, switch to the Security Rule tab and add the new security handler in the Custom Security Handler field.

Data Sets Page - Security Tab



Copy an Oracle Enterprise Command Center Framework Page

An administrator must copy the required dashboard and then add new components to change the location order of them. Use the newly- added attributes in the component configuration.

See: Extending an Existing Dashboard, page 8-2 for more details about the Copy page and adding/configuring new components.

Create a New FND Function

After creating the page, the Administrator has to create an Oracle Application Object Library function, FND_FORM_FUNCTION, to show the newly created page inside Oracle E-Business Suite. See: Create FND Function, page 8-22.

Oracle E-Business Suite Personalization

Update Oracle E-Business Suite through personalization with the URL for the newly created page. See: Personalize the Oracle Application Framework Page, page 8-25.

From Oracle E-Business Suite, verify that a user can open the new page and the new UI is shown. If everything is working fine, then export the application to back up and propagate your changes to other environments.

Back Up an Updated Application

Test and create a new export of application to back up and propagate your changes to other environments. See: Test and Export, page 8-26.

Create a New Data Set

Based on the flexible structure of Oracle Enterprise Command Center Framework, you can create your data set and build a custom dashboard on top of it based on your business requirements.

The steps below summarize the process of creating a new data set, adding it to a new application, building a new custom dashboard, and accessing it from an Oracle E-Business Suite functional responsibility.

1. Create a new PL/SQL package for data load. See Sample PL/SQL Package, page A-2 for information on creating a PL/SQL package.

2. Create a new data set. Refer to Data Set, page 5-1 for details about how to create new data.

3. Create data set metadata. See Metadata, page 5-4 and Dynamic Metadata, page 6-1 for more details about metadata.

4. Create a new application or use an existing application and assign the data set to the application. See Application, page 5-17 for more details about creating an application.

5. Using the Data Load Submission option from the Oracle Enterprise Command Center Framework Administrator UI, run the full load. Refer to Administration of Enterprise Command Centers, *Oracle E-Business Suite Administering Enterprise Command Centers* for more information.

6. Create a new page in the new/existing application. For more information on creating a new page, see Page, page 5-17.

7. Design the page using the different ECC visualization components. See Anatomy of Oracle Enterprise Command Center Framework UI Components, page 7-1 for more details about how to add and configure visualization components.

After designing the dashboard, you need to integrate this new dashboard with Oracle E-Business Suite, using the steps below.

1. Create a security handler. Refer to Sample Custom Security Handler, page E-2 for more details about security handlers.

2. In Oracle E-Business Suite, create a new function (FND function). See Create a New FND Function, page 8-35 for information about creating functions.

3. Create a new permission set or use the existing one. See Permission Sets, *Oracle E-Business Suite Security Guide* for more information.

4. Configure security for the data set. In the Oracle Enterprise Command Center Framework Administrator UI, select the Data Set option and:

    1. Enable security.

    2. Add the security handler name.

    3. Add the privilege (FND function).

5. Back up the application. See Back Up an Updated Application, page 8-36 for more details about application export.

6. Within Oracle E-Business Suite, create a new Oracle Application Framework page, or add a new tab to an existing Oracle Application Framework page. See Add a New Tab to a Command Center Oracle Application Framework Page, page 8-28 for more details about adding a new tab.

7. Create a concurrent program to be used to run the full load directly from Oracle E-Business Suite. See the Sample Concurrent Program, page C-2 section for more information. Also refer to Overview of Concurrent Programs and Requests, *Oracle E-Business Suite Setup Guide* and Overview of Concurrent Processing, *Oracle E-Business Suite Developer's Guide* for details on concurrent programs.

9

Personalization Template

Introduction

The ECC Personalization Template is a seeded Oracle Application Framework (OAF) page shipped as part of the ECC Developer responsibility. A seeded FND function is provided in which customers can easily add and remove tabs within this seeded OAF page. This feature allows users to access and manage multiple dashboards from a single responsibility so that they can have a full view of all business operations.

By default, only one tab is displayed in the page. There are four hidden tabs. To display the hidden tabs, the developer has to set the rendered property for the respective tab heading. r.

Administrators can extend Oracle Enterprise Command Centers for the following use cases:

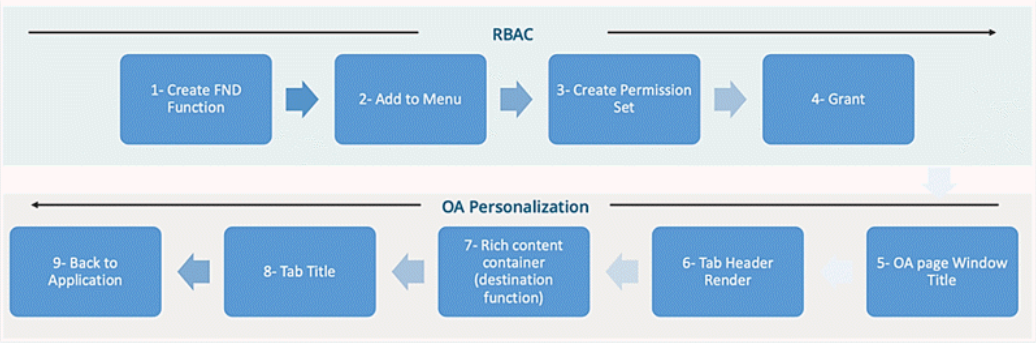
- Add new dashboards to existing dashboards
- Create new command centers

Implementation Showcase: Adding New Dashboards to Existing Dashboards

This section describes an example of providing access to the Local Agreements dashboard to the users who have access to the Purchasing, Vision Operations responsibility.

Personalization Steps

Personalization Steps



The Role Based Access Control (RBAC) steps to use the Personalization Template are:

1. Create a new FND function.
2. Add the function to a menu.
3. Create a new permission set.
4. Create a new access role and a new grant.

The Oracle Application Framework Personalization steps are:

1. Update the personalization template OAF page window title.
2. Set the "rendered" property to 'true' in Tab Header sections.
3. Add the ECC Page FND function to the respective rich container (destination function).
4. Add the tab title.
5. Return to the application.

Prerequisites: Extend the Existing Agreements Dashboard

In this example, before performing the above steps, the following prerequisites are done to extend the existing Agreements dashboard:

1. Create a new data set.
2. Create metadata attributes.

3. Assign this new data set to the Procurement application and run the data load process.

4. Copy the existing Agreements dashboard and extend it by using the newly-created data set instead of the existing one. Include the new attributes.

5. Create a data load package and security handler.

Role Based Access Control Steps

Define a function:

1. Log in to Oracle E-Business Suite with administrator credentials.

2. Create a new FND Function for the Local Agreements Dashboard.

    1. Navigate to the Functional Administrator responsibility, then choose Core Services, then Functions.

    2. Search for the code: `PO_PCC_ECC_AGREEMENTS`

3. Click on the **Duplicate** icon displayed for `PO_PCC_ECC_AGREEMENTS`.

4. Change the details in the duplicate function page as described below:

    1. Name: `PO PCC ECC Local Agreements Page`

    2. Code: `XX_PO_PCC_ECC_LOCAL_AGREEMENTS`

    3. Click the **Continue** button.

    4. Change the value for HTML Call: `GWY.jsp?targetAppType=ECC&targetPage=web/eccapp/po_pcc/xx-pcc-local-agreements`

    5. Click the **Submit** button.

5. Add the ECC Personalization Template to the Procurement Command Center menu:

    1. Navigate to Functional Administrator Responsibility, then Core Services, and then choose Menus.

    2. Search for the code: `PO_PCC_MAIN`.

6. Click the **Update** button for the "Procurement Command Center" menu.

7. In the menu manager section, click on the "+" icon to add the following menu entry details:
    - Prompt: `Local Agreements`
    - Function: `ECC Personalization Template`
8. Click **Apply** to save the menu.
9. Create a Permission Set for Local Agreements.
    1. Navigate to the Functional Administrator Responsibility, then choose Security, and then Permission Sets.
    2. Click the **Create Permission Set** button.
    3. Provide the details below in the Create Permission Set page:
        - Name: `PO PCC Local Agreements Permission Set`
        - Code: `PO_PCC_LOCAL_AGREEMENTS_PS`
    4. In the Permission Builder section, click the '+' icon to add the following permission:
        - Permission: `PO PCC ECC Local Agreements page`
    5. Click **Apply** to create the permission set.
10. Create a grant for Local Agreements:
    1. Navigate to the Functional Administrator Responsibility, then choose Security, and then Grants.
    2. Click the **Create Grant** button.
    3. Provide the details below:
        - Name: `Procurement Local Agreements Grant`
        - Grantee Type: `Group of Users`
        - Grantee: `PO PCC ECC Role`
    4. Choose the responsibility **Purchasing, Vision Operations (USA)**.
    5. Click the **Next** button.

6. For Set, specify `PO PCC Local Agreements Permission Set`.

7. Click the **Next** button and then the **Finish** button.

11. Clear the application cache.

    1. Navigate to the Functional Administrator responsibility, then choose Core Services, then Caching Framework, then Global Configuration.

    2. Click the **Clear All Cache** button and confirm that you want to clear the cache.

OAF Personalization: Personalize ECC Personalization Template to Add Local Agreements Dashboard

In this example, the steps in Oracle Application Framework are described below.

Personalize ECC Personalization Template to add ECC Procure to Pay dashboards:

1. Log in to Oracle E-Business Suite with administrator credentials.

2. Navigate to Purchasing, Vision Operations (USA), then Procurement Command Center, then Local Agreements.

3. Click the Oracle E-Business Suite **Settings** icon.

4. Select the **Personalize Page** option.

5. In the Personalization structure table, click the **Personalize** icon for the Page Layout section to update the window title.

6. Set the window title to "Local Agreements" and click on **Apply**.

7. Select the **Personalize** option for the rich container (`dashboardRN1`).

8. Update with the rich container with the details below:

    - Destination Function: `XX_PO_PCC_ECC_LOCAL_AGREEMENTS`

    - Rendered: **true**

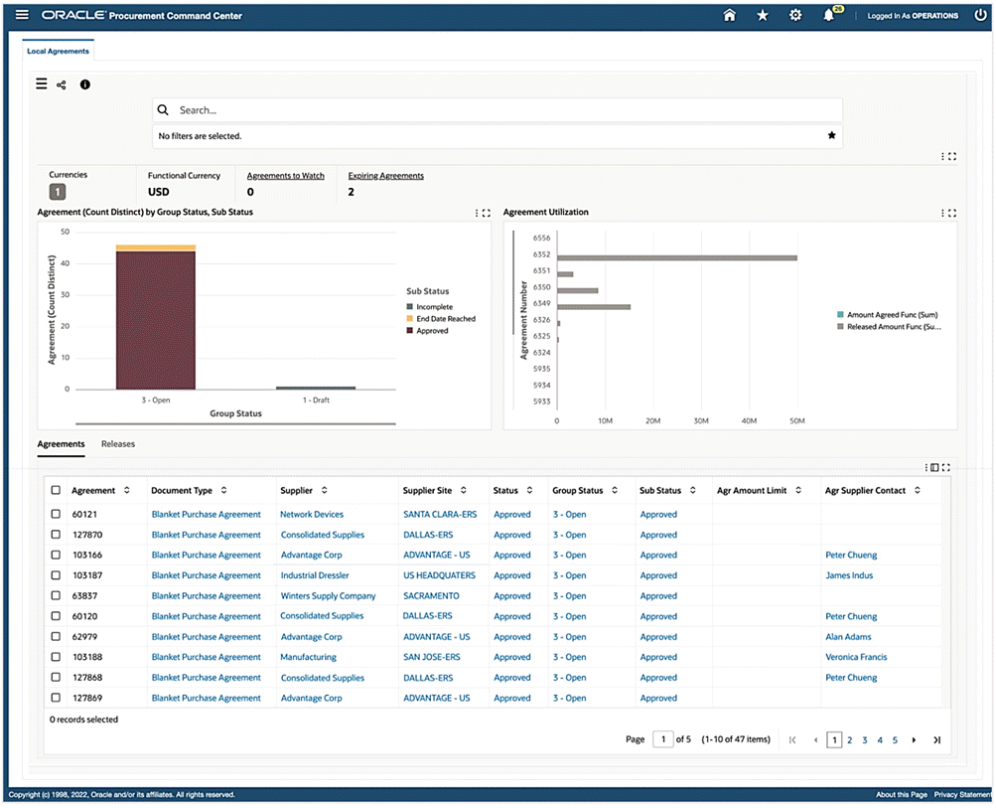
    - Title: `Local Agreements`

9. Set the subtab title by clicking on the **Personalize** icon for the first link under Sub Tabs.

10. Update the Text property to `Local Agreements`.

11. Click on **Return to Application** to access the dashboards.

Local Agreements Dashboard



To add other shipped dashboards, such as the Agreements dashboard or Requisitions dashboard in the same page, follow the same personalization process for other subtabs available to the ECC personalization template.

1. Select the "**Personalize** option under **Settings**.

2. Personalize the header (`dashboardHdrRN2`)

3. Set the **Rendered** property to `true`.

4. Personalize the Rich Container (`dashboardRN2`).

5. Update the details below.

   - Destination Function: `PO_PCC_ECC_REQUISITIONS`

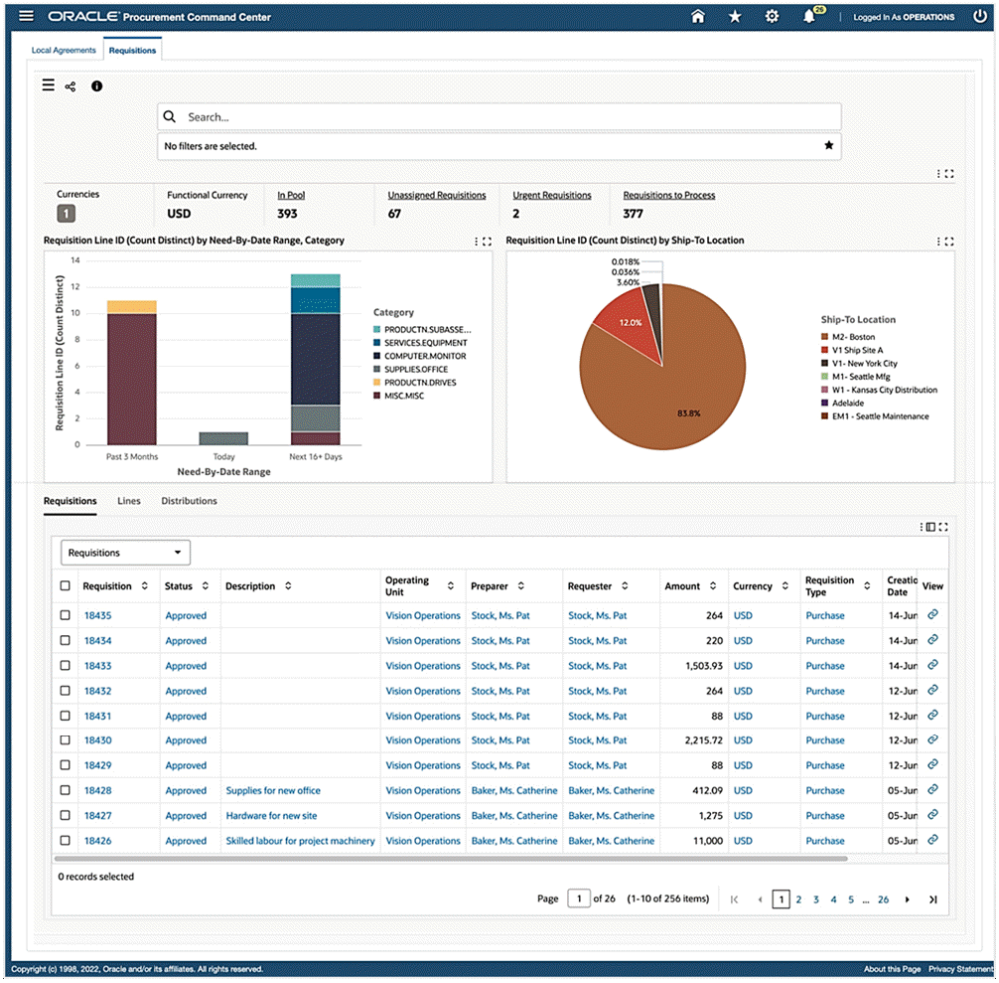
   - Rendered: **true**

- Title: `Requisitions`

Click **Apply**.

6. Set the subtab title by selecting the **Personalize** icon for the second Sub Tab link.
7. Update the **Text** property to `Requisitions`.
8. Click on **Return to Application** to access the dashboard.

Requisitions Dashboard



10

Implementing the Push Model for Loading Data

Overview

Oracle Enterprise Command Center Framework supports real-time updates to the dashboard data using a special model of data load: Push Load. Changes in the source system are immediately captured by the Push model by triggering a data load.

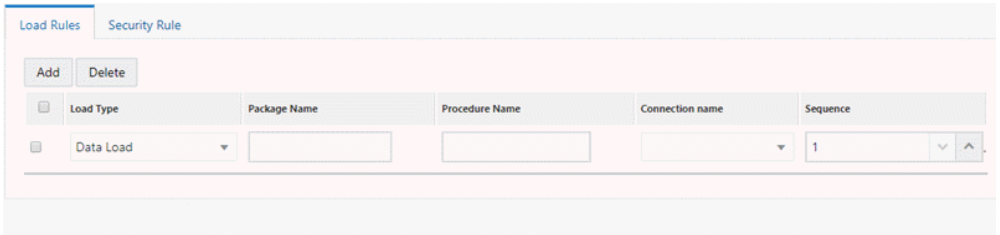
The Push model is essential for business use cases that have high-frequency transactions and need to render the dashboards with up-to-date information.

The Push model skips concurrent processing overhead and allows business users to leverage near real-time data so they can make quicker and prompt decisions. An asynchronous HTTP POST Call is required to perform a PUSH request.

Implementing Push Load (Using ECC Client JAVA API)

1. Create the data set with load type: "Data Load".

Example of Creating a Data Set



2. Do not include any package or procedure for the data set.

3. Create the `PushRequestHeader` object for generating the ECC Client connection.

```
PushRequestHeader pushRequestHeaderObj = new PushRequestHeader();
     pushRequestHeaderObj.setSystemName(SystemName.EBS);
     pushRequestHeaderObj.setAdditionalParams(additionalParams);
     pushRequestHeaderObj.setCookies(cookies);
     pushRequestHeaderObj.setCustomLoadParams(null);
     pushRequestHeaderObj.setDatasetKey(eccDataset);
     pushRequestHeaderObj.setEccAppShortName(eccAppName);
     pushRequestHeaderObj.setEccServiceBaseURL(eccWebEntyUrl);
     pushRequestHeaderObj.setExternalRequestId(extnRequestId);
     pushRequestHeaderObj.setLastRunDate(new Date());
     pushRequestHeaderObj.setLoadType(loadType);
     pushRequestHeaderObj.setLogLevel(logLevel);
     pushRequestHeaderObj.setLanguages(pLanguages);
     pushRequestHeaderObj.setResetCollection(resetCollectionFlag);
```

4. The `EccPushClient` object is required to push data. Use the call: `EccClientUtil.getPushClientService(SystemName.EBS).getPushClient` to get the `eccPushClient` object.

5. Ways to generate a Push request:

    1. Push request using Java objects as the source during a transaction:

        1. Identify the set of attribute keys for the records to be sent by the PUSH request and create the RecordSet object.

        2. Identify the operation to be performed for the set of records. Possible values are the following using LoadOperationType Enum: UPDATE, DELETE, INSERT, UPSERT, REPLACE.

    2. Use of a Result Set for a batch operation:

        Identify the result set and give the following call passing loadOperation and batchSize of the records to be process in a job. You also need to create a GFMGenerator class object so that gfmToken can be created by the ECC Client to post each batch request.

Tracking data load status

The data load status can be checked using the ECC Client method call, passing the job ID retrieved from the response of the data load request.

Notes for Developers

1. Please ensure that the `ecc-client.zip` file and the Jackson libraries from (`/u01/R122_EBS/fs1/EBSapps/comn/java/lib`) from the central environment are in the build path of the code changes for compilation. The files are listed below:

```
/u01/R122_EBS/fs1/EBSapps/comn/java/lib/jackson_annotations.zip
/u01/R122_EBS/fs1/EBSapps/comn/java/lib/jackson_core.zip
/u01/R122_EBS/fs1/EBSapps/comn/java/lib/jackson_databind.zip
/u01/R122_EBS/fs1/EBSapps/comn/java/lib/ecc-client.zip
```

2. Import the following classes into the client call:

```
import oracle.ecc.client.Cookie;
import oracle.ecc.client.SystemName;
import oracle.ecc.client.impl.EccPushClient;
import oracle.ecc.client.model.LoadOperationType;
import oracle.ecc.client.model.LoadType;
import oracle.ecc.client.model.LogLevel;
import oracle.ecc.client.model.PushRequestHeader;
import oracle.ecc.client.model.ResponseHeader;
import oracle.ecc.client.util.EccClientUtil;
import oracle.ecc.client.util.GFMGenerator;
```

3. To generate `GFMToken`, use the following code:

```
GFMGenerator gfmGeneratorObj = new GFMGenerator(connection);
String gfmToken = gfmGeneratorObj.generateNewGFMToken();
```

4. To get the list of supported ECC languages, use the following function and class: method:

```
ArrayList<String> getSupportedLanguages(String eccServiceBaseURL,
Map<String, List<String>> additionalParams, SystemName systemName,
LogLevel logLevel)
class: oracle.ecc.client.util.EccClientUtil
```

   Note that`gfmToken` needs to be added to `additionalParams`as described in the table below.

5. Define attributes of `PushRequestHeader`:

Attributes of `PushRequestHeader`

Parameter	Mandatory	Meaning	Possible Values	Impact
SystemName	Yes	The Source System of PUSH Request	EBS	It is important to identify that the request is from the Oracle E-Business Suite system, and the request will be verified using the Oracle E-Business Suite Security Manager.

Parameter	Mandatory	Meaning	Possible Values	Impact
externalRequestID	Yes	The Source ID of the push request	Long value up to 100 characters	This will be useful for auditing requests from a source.
eccAppShortName	Yes	ECC Application Short Name	Any string matching the ECC application short name	The Push request is executed for the respective application data set.
datasetKey	Yes	ECC Dataset Key	Any data set key of the application identified above	The Push request is executed for the respective data set.
loadType	Yes	Identifies if the load is PUSH or INCREMENTAL OR FULL	LoadType. DATA_LOAD	The value `DATA_LOAD` ensures that the load request is PUSH.
eccServiceBaseURL	Yes	This value is ECC ENTRY URL which is /ecc appended to the Oracle E-Business Suite web entry URL.	The value of the Oracle E-Business Suite entry URL can be identified using profile value "APPS_FRAMEWORK_AGENT"	Sends the request to the appropriate URL.
customLoadParams	No	This is made available to send other parameters to the job.	null	Not in use currently.

Parameter	Mandatory	Meaning	Possible Values	Impact
cookies	Yes (if no gfmToken used)		EBS Session cookie	Security
additionalParams	Yes (if no cookies sent)	Can be used to pass additional parameters to the URL	(See below)	Security
logLevel	No	Logging level for ECC data load request	LogLevel Enum in ECC Client Default value: ERROR	
pLanguages	No	Languages for which data is sent	EBS Session language for which the PUSH request update is triggered Default value: "US"	
lastRunDate	Yes (For Full Load and Incremental Load)	The last run date from which the data has been sent for DATA LOAD		This ensures that the last run date is updated in ECC for the data set.
resetCollectionFlag	Yes (For Full Load only)		true, false	The data set collection is reset and only the new data will be available after the data load.

A possible value for `additionalParams` is:

```
gfmToken
Map<String, List<String>> additionalParams = new HashMap<String,
List<String>>();
if (null != gfmToken)
additionalParams.put("gfmString", Arrays.asList(gfmToken));
```

Code Samples

1. Code sample for `PushRequestHeader`:

```
EccPushClient eccPushClientObj = EccClientUtil.getPushClientService
(SystemName.EBS).getPushClient(pushRequestHeaderObj);
```

2. Code sample for Push Request using Java Objects as source:

```
List<Map<String,Object>> recordList = new ArrayList<Map<String,
Object>>();
List<String> attributeKeyList = new ArrayList<String>();
attributeKeyList.add("ECC_SPEC_ID");
attributeKeyList.add("BRANCH");
attributeKeyList.add("CLASS");
attributeKeyList.add("STUDENT_NAME");
/* US_COURSES is a translated attribute for language US*/
attributeKeyList.add("US_COURSES");
attributeKeyList.add("JOIN_DATE");
Map<String,Object> record = new HashMap<String,Object>();
record.put("ECC_SPEC_ID", "2");
record.put("BRANCH", "CSC");
record.put("CLASS", "2");
record.put("STUDENT_NAME", "RAJNI KANTH");
/* COURSES is multi-value attribute. Hence values are '|' separated
*/
record.put("US_COURSES", "CS|ICT");
/* dates should be send in canonical format */
record.put("JOIN_DATE", new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:
ss'Z'").parse("2018-01-01T00:00:00Z"));
recordList.add(record);

RecordSet recordSetObj= EccClientUtil.createRecordSet
(attributeKeyList,recordList);
recordSetObj.setOperation(LoadOperationType.REPLACE);
ResponseHeader responseHeaderObj = eccPushClientObj.pushData
(recordSetObj);
```

3. A code sample for the Use of Result Set for a batch operation:

```
List<ResponseHeader> responseHeaderList = eccPushClientObj.pushData
(LoadOperationType.REPLACE, rs, 1000,gfmGeneratorObj);
```

   The recommended size of a batch job is 1000.

4. Sample SQL query to generate the result set (similar to the query for PULL model):

```
select * from (select ECC_SPEC_ID,BRANCH,CLASS,STUDENT_NAME,COURSES,
JOIN_DATE,LANGUAGE_CODE
from ECC_CLIENT_ENROL_TEST ) t
PIVOT ( max(COURSES) as COURSES for LANGUAGE_CODE in ('US' \"US\"))
```

   - If the `resetCollection` is true, then `lastRunDate`is required. Else, the request will fail.
   - If `etl_languages` is null when `resetCollection` is true, it defaults to the base language.

- If `reset_collection_flag` is set to true, then it is similar to Full Load (PUSH).
- If `reset_collection_flag` is set to false(default) and `last_run_date` is defined, it is the same as Incremental Load Push.

A

Sample PL/SQL Package

Example from Oracle iProcurement

```
CREATE OR REPLACE PACKAGE BODY icx_ecc_util_pvt AS
  /* $Header: ICXECCOP.pls 120.0.1 2019/02/28 10:13:01 noship $ */
  /*  Utility Package for populating ECC metadata. */
  FUNCTION get_descriptors_sql(
      p_category IN NUMBER,
      p_load_type IN VARCHAR2,
      p_languages IN VARCHAR2,
      p_ds_last_success_run IN TIMESTAMP)
    RETURN CLOB
  IS
    l_sql_text CLOB;
    v_for_lang_pivot_clause varchar2(400) := FND_ECC_UTIL_MLS_PVT.
GEN_ECC_MLS_PIVOT_FOR_LANG_CL(p_languages);
    l_desc_sql_text VARCHAR2(30000) := ' select * from ICX_CAT_ECC_' ||
p_category || '_VL';
    l_pivot_pre_sql varchar2(2000) := 'select * from ( ';
    l_pivot_post_sql1 VARCHAR2(2000) := ' )
              PIVOT ( count(1) as IS_TRANSLATION_AVAILABLE ';
    l_pivot_post_sql2 VARCHAR2(2000) := ' ';
    l_pivot_post_sql3 VARCHAR2(2000) := 'FOR LANGUAGE IN
('||v_for_lang_pivot_clause||'))'  ;
    l_last_update_condition varchar2(1000) := ' WHERE
ECC_LAST_UPDATE_DATE >=  to_date(to_char(to_timestamp
('||''''||p_ds_last_success_run||''''||'),''DD-MON-YY HH24.MI.
SS''),''DD-MON-YY HH24.MI.SS'')';
    l_trans_attributes     dbms_sql.varchar2_table;
    cursor c_trans_attributes  is
      SELECT Decode( UPPER(regexp_replace(KEY,'[-*# ]','_') ), 'SIZE',
'SIZE_', UPPER(regexp_replace(KEY,'[-*# ]','_') ) ) ATTRIBUTE_NAME
      FROM   ICX_CAT_ATTRIBUTES_TL
      WHERE stored_in_table ='PO_ATTRIBUTE_VALUES_TLP'
      AND STORED_IN_COLUMN IS NOT NULL
      AND ATTRIBUTE_ID>28
      AND LANGUAGE='US'
      AND RT_CATEGORY_ID=p_category;
  BEGIN
    OPEN c_trans_attributes;
    FETCH c_trans_attributes BULK COLLECT INTO l_trans_attributes;
    CLOSE c_trans_attributes;
/*
    IF(l_trans_attributes.COUNT = 0) THEN
      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_desc_sql_text;
      ELSE
        l_sql_text := l_desc_sql_text || l_last_update_condition;
      END IF;
      RETURN l_sql_text;
    END IF;
  */
    FOR i IN 1..l_trans_attributes.count
    LOOP
      l_pivot_post_sql2 := l_pivot_post_sql2 || ',  MAX( ' ||
l_trans_attributes(i) || ' ) as ' || l_trans_attributes(i) ;
    END LOOP;

      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_pivot_pre_sql || l_desc_sql_text ||
l_pivot_post_sql1 || l_pivot_post_sql2 || l_pivot_post_sql3;
      ELSE
        l_sql_text := l_pivot_pre_sql || l_desc_sql_text ||
l_last_update_condition || l_pivot_post_sql1 || l_pivot_post_sql2 ||
l_pivot_post_sql3;
```

```
END IF;

    RETURN l_sql_text;
  END get_descriptors_sql;
/*
** GET_ECC_DATA_LOAD_INFO - Retrieves the data load details for the
given data set key
**
** IN parameters
**  p_dataset_id - data set key
**  p_load_type - Indicating Full or incremental load (F/I/Custom).For
custom there are no data load rules defined
**  p_ds_last_success_run - Returns the last successful etl run for the
data set key
**  OUT parameters
**  x_ecc_ds_meta_rec - ecc_ds_meta_rec
**  x_return_status - return status
*/
--------------------------------------------------------------------------
  PROCEDURE get_ecc_data_load_info(
      p_dataset_key         IN VARCHAR2,
      p_load_type           IN VARCHAR2,
      p_ds_last_success_run IN TIMESTAMP,
      p_languages           IN VARCHAR2,
      p_addl_params         IN ecc_sec_field_values,
      x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec,
      x_return_status OUT NOCOPY VARCHAR2 )
  IS
    query_det_arr ecc_query_det_arr_type     := ecc_query_det_arr_type
(NULL);
    l_apps_schema_name varchar2(25) := 'APPS';
    CURSOR c_category_view
    IS
     SELECT DISTINCT RT_CATEGORY_ID FROM ICX_CAT_CATEGORIES_TL
     WHERE LANGUAGE='US'
     AND EXISTS
        (SELECT 1 FROM ALL_VIEWS
         WHERE VIEW_NAME = 'ICX_CAT_ECC_'||RT_CATEGORY_ID||'_VL'
         AND  owner=l_apps_schema_name);

    l_categories dbms_sql.number_table;
    l_category  NUMBER;
    l_sql_index number := 1;
  BEGIN
   select oracle_username
        into l_apps_schema_name
        from fnd_oracle_userid
        where read_only_flag = 'U';

    IF ( p_load_type = 'INCREMENTAL_LOAD' ) THEN
      query_det_arr(l_sql_index)     := ecc_query_det_rec(get_sql_text
(p_dataset_key,p_load_type,g_icx_ecc_del_op, 'ITEMS',
p_ds_last_success_run,p_languages),g_icx_ecc_del_op);
      query_det_arr.extend ();
      l_sql_index := l_sql_index +1;
    END IF;

    query_det_arr(l_sql_index) := ecc_query_det_rec(get_sql_text
(p_dataset_key,p_load_type,g_icx_ecc_ins_op, 'ITEMS',
p_ds_last_success_run,p_languages),g_icx_ecc_ins_op);

    query_det_arr.extend ();
    l_sql_index := l_sql_index +1;
    query_det_arr(l_sql_index) := ecc_query_det_rec(get_sql_text
(p_dataset_key,p_load_type,g_icx_ecc_upd_op, 'HIERARCHY',
p_ds_last_success_run,p_languages),g_icx_ecc_upd_op);
```

```
    query_det_arr.extend ();
    l_sql_index := l_sql_index +1;
    query_det_arr(l_sql_index) := ecc_query_det_rec(get_sql_text
(p_dataset_key,p_load_type,g_icx_ecc_upd_op, 'ZONESB',
p_ds_last_success_run,p_languages),g_icx_ecc_upd_op);

    query_det_arr.extend ();
    l_sql_index := l_sql_index +1;
    query_det_arr(l_sql_index) := ecc_query_det_rec(get_sql_text
(p_dataset_key,p_load_type,g_icx_ecc_upd_op, 'ZONESP',
p_ds_last_success_run,p_languages),g_icx_ecc_upd_op);

    query_det_arr.extend ();
    l_sql_index := l_sql_index +1;
    query_det_arr(l_sql_index) := ecc_query_det_rec(get_sql_text
(p_dataset_key,p_load_type,g_icx_ecc_upd_op, 'ZONESI',
p_ds_last_success_run,p_languages),g_icx_ecc_upd_op);


    OPEN c_category_view;
    FETCH c_category_view BULK COLLECT INTO l_categories;
    CLOSE c_category_view;
    FOR i IN 1..l_categories.count
    LOOP
      query_det_arr.extend ();
      l_sql_index := l_sql_index +1;
      query_det_arr(l_sql_index) := ecc_query_det_rec
(get_descriptors_sql(l_categories(i), p_load_type, p_languages,
p_ds_last_success_run ),g_icx_ecc_upd_op);
    END LOOP;
    x_ecc_ds_meta_rec := ecc_ds_meta_rec(p_dataset_key,query_det_arr);
    x_return_status   := 'S';
  END get_ecc_data_load_info;
  FUNCTION get_sql_text(
      p_dataset_key         IN VARCHAR2,
      p_load_type           IN VARCHAR2,
      p_operation           IN VARCHAR2,
      p_data_type           IN VARCHAR2,
      p_ds_last_success_run IN TIMESTAMP,
      p_languages           IN VARCHAR2)
    RETURN CLOB
  IS
    l_sql_text CLOB;
    v_for_lang_pivot_clause varchar2(400) := FND_ECC_UTIL_MLS_PVT.
GEN_ECC_MLS_PIVOT_FOR_LANG_CL(p_languages);
    l_last_update_condition varchar2(1000) := ' WHERE
ECC_LAST_UPDATE_DATE >=  to_date(to_char(to_timestamp
('||''''||p_ds_last_success_run||''''||'),''DD-MON-YY HH24.MI.
SS''),''DD-MON-YY HH24.MI.SS'') ';
    l_pivot_pre_sql VARCHAR2(100) := 'SELECT * FROM  ( ';
    l_pivot_post_sql VARCHAR2(2000) := ' )
              PIVOT (
                MAX(SHOPPING_CATEGORY) AS SHOPPING_CATEGORY,
                MAX(DESCRIPTION) AS DESCRIPTION,
                MAX(UNIT_OF_MEASURE) AS UNIT_OF_MEASURE,
                MAX(COMMENTS) AS COMMENTS,
                MAX(LONG_DESCRIPTION) AS LONG_DESCRIPTION,
                MAX(ITEM_SOURCE_TEXT) AS ITEM_SOURCE_TEXT,
                MAX(CONTENT_TYPE_FILTER) AS CONTENT_TYPE_FILTER,
                MAX(DISPLAY_PRICE) AS DISPLAY_PRICE,
                MAX(PRICE_BREAK) AS PRICE_BREAK,
                MAX(ITEM_RATING_FILTER) AS ITEM_RATING_FILTER,
                MAX(SUPPLIER_RATING_FILTER) AS SUPPLIER_RATING_FILTER,
                MAX(PUNCHOUT_MORE_DETAILS) AS PUNCHOUT_MORE_DETAILS,
                MAX(CONTENT_TYPE_RESULTS) AS CONTENT_TYPE_RESULTS,
                MAX(SHOPPING_CATEGORY_1) AS SHOPPING_CATEGORY_1,
```

```
MAX(SHOPPING_CATEGORY_2) AS SHOPPING_CATEGORY_2,
                MAX(SHOPPING_CATEGORY_3) AS SHOPPING_CATEGORY_3,
                MAX(KEYWORDS) AS KEYWORDS,
                MAX(SOURCE) AS SOURCE,
                COUNT(1) AS IS_TRANSLATION_AVAILABLE
                FOR LANGUAGE IN ('||v_for_lang_pivot_clause||'))'  ;
    l_items_sql_text VARCHAR2(30000) :=
    'SELECT   ECC_SPEC_ID,
              THUMBNAIL_IMAGE          ,
              SHOPPING_CATEGORY        ,
              SUPPLIER                 ,
              SUPPLIER_SITE            ,
              SUPPLIER_PART_NUM        ,
              SUPPLIER_PART_AUXID      ,
              INTERNAL_ITEM_NUM        ,
              SOURCE                   ,
              MANUFACTURER             ,
              MANUFACTURER_PART_NUM    ,
              PURCHASING_CATEGORY      ,
              DESCRIPTION              ,
              ITEM_REVISION            ,
              UNIT_OF_MEASURE          ,
              PRICE                    ,
              CURRENCY                 ,
              FUNCTIONAL_PRICE         ,
              FUNCTIONAL_CURRENCY      ,
              AVAILABILITY             ,
              LEAD_TIME                ,
              UNSPSC                   ,
              ALIAS                    ,
              COMMENTS                 ,
              LONG_DESCRIPTION         ,
              ATTACHMENT_URL           ,
              SUPPLIER_URL             ,
              MANUFACTURER_URL         ,
              ORG_ID                   ,
              LANGUAGE                 ,
              ITEM_SOURCE_TEXT         ,
              CONTENT_TYPE_FILTER      ,
              dbms_lob.substr(ATTACHMENT, 4000, 1) ATTACHMENT  ,
              ZONE_ID                  ,
              DISPLAY_PRICE            ,
              CONTENT_ID               ,
              CONTENT_URL              ,
              KEYWORDS                 ,
              CONTENT_TYPE             ,
              HIDE_SMARTFORM_ATTRS     ,
              HIDE_INFOCONTENT_ATTRS   ,
              PRICE_BREAK              ,
              HIDE_PRICE_BREAK         ,
              ITEM_RATING_FILTER       ,
              SUPPLIER_RATING_FILTER   ,
              HIDE_ADDTOCART           ,
              PUNCHOUT_MORE_DETAILS    ,
              OPEN_DESCRIPTION_IN_NEW_TAB ,
              ITEM_RATING              ,
              SUPPLIER_RATING          ,
              HIDE_CONTENT_TYPE        ,
              CONTENT_TYPE_RESULTS     ,
              SHOPPING_CATEGORY_1,
              SHOPPING_CATEGORY_2,
              SHOPPING_CATEGORY_3,
              ZONESB,
              ZONESP,
              ZONESI,
              PREFERRED_SUPPLIER_ICON,
```

```
ECO_FRIENDLY_SUPPLIER_ICON,
              GREEN_SUPPLIER_ICON,
              OFF_CONTRACT_ITEM_ICON,
              OVER_PRICED_ICON
              FROM ICX_CAT_ECC_ITEMS_V ' ;
    l_hierarchy_sql_text_pre VARCHAR2(30000) := 'SELECT * FROM(
              SELECT
              ecc_spec_id,
              shopping_category,
              CASE
              WHEN shopping_category_3 IS NOT NULL THEN
shopping_category_3
              WHEN shopping_category_2 IS NOT NULL THEN
shopping_category_2
              ELSE ''Others'' END AS shopping_category_1 ,
              CASE
              WHEN shopping_category_3 IS NOT NULL AND
shopping_category_2 IS NOT NULL  THEN shopping_category_2
              WHEN  shopping_category_2 IS NOT NULL AND
shopping_category_1 IS NOT NULL  THEN shopping_category_1
              ELSE shopping_category_1
              END AS shopping_category_2,
              CASE
              WHEN shopping_category_3 IS NOT NULL AND
shopping_category_2 IS NOT NULL  AND shopping_category_1 IS NOT NULL
THEN shopping_category_1
              ELSE null
              END AS shopping_category_3,
              LANGUAGE
              FROM  icx_ecc_category_hierarchy ';
    l_hierarchy_sql_text_post VARCHAR2(30000) := ' ) PIVOT(
                MAX(SHOPPING_CATEGORY) AS SHOPPING_CATEGORY,
                MAX(SHOPPING_CATEGORY_1) AS SHOPPING_CATEGORY_1,
                MAX(SHOPPING_CATEGORY_2) AS SHOPPING_CATEGORY_2,
                MAX(SHOPPING_CATEGORY_3) AS SHOPPING_CATEGORY_3
                FOR LANGUAGE IN ('||v_for_lang_pivot_clause||'))' ;
    l_zonesb_sql_text      VARCHAR2(30000) := 'SELECT
              ecc_spec_id,
              zonesb
              from icx_cat_ecc_zones_b';
    l_zonesp_sql_text      VARCHAR2(30000) := 'SELECT
              ecc_spec_id,
              zonesp
              from icx_cat_ecc_zones_p';
    l_zonesi_sql_text      VARCHAR2(30000) := 'SELECT
              ecc_spec_id,
              zonesi
              from icx_cat_ecc_zones_i';

  BEGIN

    IF ( p_operation = g_icx_ecc_del_op ) THEN
      l_sql_text       := 'SELECT recordkey ECC_SPEC_ID
                           FROM icx_cat_endeca_item_attributes
                           WHERE attributekey LIKE ''##DELETERECORD##'''
;
      return l_sql_text;
    END IF;

    IF ( p_data_type     = 'ITEMS' ) THEN
      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_pivot_pre_sql || l_items_sql_text ||
l_pivot_post_sql;
      ELSE
        l_sql_text := l_pivot_pre_sql || l_items_sql_text ||
l_last_update_condition || l_pivot_post_sql;
```

```
END IF;

    ELSIF( p_data_type     = 'HIERARCHY' ) THEN
      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_hierarchy_sql_text_pre ||
l_hierarchy_sql_text_post ;
      ELSE
        l_sql_text := l_hierarchy_sql_text_pre ||
l_last_update_condition || l_hierarchy_sql_text_post ;
      END IF;
    ELSIF( p_data_type     = 'ZONESB' ) THEN
      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_zonesb_sql_text;
      ELSE
        l_sql_text := l_zonesb_sql_text ||  l_last_update_condition ;
      END IF;
    ELSIF( p_data_type     = 'ZONESP' ) THEN
      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_zonesp_sql_text;
      ELSE
        l_sql_text := l_zonesp_sql_text ||  l_last_update_condition ;
      END IF;
    ELSIF( p_data_type     = 'ZONESI' ) THEN
      IF ( p_load_type     = 'FULL_LOAD' ) THEN
        l_sql_text := l_zonesi_sql_text;
      ELSE
        l_sql_text := l_zonesi_sql_text ||  l_last_update_condition ;
      END IF;
    END IF;

    RETURN l_sql_text;
  END get_sql_text;
  PROCEDURE create_ecc_desc_view(
      p_category_id IN NUMBER )
  IS
    l_view_sql       VARCHAR2(20000);
    l_category_id    NUMBER         := p_category_id;
    l_attr_list      VARCHAR2(2000) := '';
    l_attr_val_list  VARCHAR2(2000) := '';
    l_attribute_name VARCHAR2(100)  := '';
    CURSOR c_base_attrs
    IS
      SELECT UPPER(regexp_replace(KEY,'[-*# ]','_') ),
        DECODE(icon_indicator,1,'decode(PAV.'
        || stored_in_column
        || ' , ''Y'', ''true'', ''false'') ','PAV.'
        || stored_in_column) value
      FROM icx_cat_attributes_tl
      WHERE rt_category_id  = l_category_id
      AND stored_in_table   = 'PO_ATTRIBUTE_VALUES'
      AND stored_in_column IS NOT NULL
      AND attribute_id      > 28
      AND language          = 'US';
    CURSOR c_base_attrs_tl
    IS
      SELECT UPPER(regexp_replace(KEY,'[-*# ]','_') ),
        'PAV_TLP'
        || '.'
        || stored_in_column value
      FROM icx_cat_attributes_tl
      WHERE rt_category_id  = l_category_id
      AND stored_in_table   = 'PO_ATTRIBUTE_VALUES_TLP'
      AND stored_in_column IS NOT NULL
      AND attribute_id      > 28
      AND language          = 'US';
    l_attrs_tbl dbms_sql.varchar2_table;
```

```
l_attr_vals_tbl dbms_sql.varchar2_table;
  BEGIN
    l_view_sql := 'create or replace view ICX_CAT_ECC_' || l_category_id
|| '_VL (';
    l_view_sql := l_view_sql || 'ECC_SPEC_ID,';
    l_view_sql := l_view_sql || 'LANGUAGE, ';
    l_view_sql := l_view_sql || 'ECC_LAST_UPDATE_DATE ';
    OPEN c_base_attrs;
    FETCH c_base_attrs BULK COLLECT INTO l_attrs_tbl,l_attr_vals_tbl;
    FOR i IN 1..l_attrs_tbl.count
    LOOP
      l_attribute_name     := l_attrs_tbl(i);
      IF ( l_attribute_name = 'SIZE' ) THEN
        l_attribute_name   := l_attribute_name || '_';
      END IF;
      l_attr_list     := l_attr_list || ' , ' || l_attribute_name;
      l_attr_val_list := l_attr_val_list || ' , ' || l_attr_vals_tbl(i);
    END LOOP;
    CLOSE c_base_attrs;
    OPEN c_base_attrs_tl;
    FETCH c_base_attrs_tl BULK COLLECT INTO l_attrs_tbl,l_attr_vals_tbl;
    FOR i IN 1..l_attrs_tbl.count
    LOOP
      l_attribute_name     := l_attrs_tbl(i);
      IF ( l_attribute_name = 'SIZE' ) THEN
        l_attribute_name   := l_attribute_name || '_';
      END IF;
      l_attr_list     := l_attr_list || ' , ' || l_attribute_name;
      l_attr_val_list := l_attr_val_list || ' , ' || l_attr_vals_tbl(i);
    END LOOP;
    CLOSE c_base_attrs_tl;
    l_view_sql         := l_view_sql || l_attr_list || ') AS ';
    l_view_sql         := l_view_sql || 'select items.inventory_item_id
|| ''#'' || items.po_line_id || ''#''
      || items.req_template_name  || ''#'' ||items.req_template_line_num
|| ''#'' || items.org_id ECC_SPEC_ID,' ;
    l_view_sql         := l_view_sql || ' items.LANGUAGE,  ';
    l_view_sql         := l_view_sql || ' GREATEST(items.
last_update_date, NVL(PAV.LAST_UPDATE_DATE, items.last_update_date), NVL
(items.last_update_date, PAV_TLP.LAST_UPDATE_DATE)) ECC_LAST_UPDATE_DATE
';
    l_view_sql         := l_view_sql || l_attr_val_list;
    l_view_sql         := l_view_sql || ' FROM
ICX_CAT_ITEMS_CTX_HDRS_TLP ITEMS, ';
    l_view_sql         := l_view_sql || ' PO_ATTRIBUTE_VALUES PAV, ' ||
' PO_ATTRIBUTE_VALUES_TLP PAV_TLP ';
    l_view_sql         := l_view_sql || ' WHERE items.inventory_item_id
= PAV.inventory_item_id(+) ';
    l_view_sql         := l_view_sql || ' AND items.po_line_id  = PAV.
po_line_id(+) ';
    l_view_sql         := l_view_sql || ' AND items.req_template_name  =
PAV.req_template_name(+) ';
    l_view_sql         := l_view_sql || ' AND items.
req_template_line_num  = PAV.req_template_line_num(+) ';
    l_view_sql         := l_view_sql || ' AND items.org_id  = PAV.org_id
(+) ';
    l_view_sql         := l_view_sql || ' AND items.inventory_item_id  =
PAV_TLP.inventory_item_id(+) ';
    l_view_sql         := l_view_sql || ' AND items.po_line_id  =
PAV_TLP.po_line_id(+) ';
    l_view_sql         := l_view_sql || ' AND items.req_template_name  =
PAV_TLP.req_template_name(+) ';
    l_view_sql         := l_view_sql || ' AND items.
req_template_line_num  = PAV_TLP.req_template_line_num(+) ';
    l_view_sql         := l_view_sql || ' AND items.org_id  = PAV_TLP.
org_id(+) ';
```

```
l_view_sql         := l_view_sql || ' AND items.language  = PAV_TLP.
language(+) ';
    IF ( l_category_id <> 0 ) THEN
      l_view_sql       := l_view_sql || ' AND items.ip_category_id = ';
      l_view_sql       := l_view_sql || l_category_id;
    END IF;
    --    dbms_output.put_line(substr(l_view_sql, 0, 200));
    --    dbms_output.put_line(substr(l_view_sql, 201, 200));
    --    dbms_output.put_line(substr(l_view_sql, 401, 200));
    --    dbms_output.put_line(substr(l_view_sql, 601, 200));
    --    dbms_output.put_line(substr(l_view_sql, 801, 200));
    EXECUTE IMMEDIATE l_view_sql;
  END create_ecc_desc_view;
  PROCEDURE create_ecc_desc_views
  IS
    l_cat_id_list dbms_sql.varchar2_table;
    CURSOR c_categories
    IS
      SELECT DISTINCT rt_category_id
      FROM icx_cat_attributes_tl
      WHERE EXISTS
        (SELECT 1 FROM icx_cat_items_ctx_hdrs_tlp WHERE
ip_category_id=rt_category_id
      UNION
      SELECT 1 FROM icx_cat_punchout_items WHERE
ip_category_id=rt_category_id
       )
      OR rt_category_id=0;
    BEGIN
      icx_ecc_util_pvt.drop_ecc_desc_views;
      OPEN c_categories;
      FETCH c_categories BULK COLLECT INTO l_cat_id_list;
      CLOSE c_categories;
      FOR i IN 1..l_cat_id_list.count
      LOOP
        icx_ecc_util_pvt.create_ecc_desc_view(l_cat_id_list(i) );
      END LOOP;
    END create_ecc_desc_views;
    PROCEDURE drop_ecc_desc_views
    IS
      l_apps_schema_name varchar2(25) := 'APPS';
      CURSOR c_ecc_views
      IS
        SELECT DISTINCT view_name
        FROM all_views
        WHERE view_name LIKE 'ICX_CAT_ECC%VL'
        AND owner=l_apps_schema_name;
      l_view_list dbms_sql.varchar2_table;
    BEGIN
        select oracle_username
        into l_apps_schema_name
        from fnd_oracle_userid
        where read_only_flag = 'U';

      OPEN c_ecc_views;
      FETCH c_ecc_views BULK COLLECT INTO l_view_list;
      CLOSE c_ecc_views;
      FOR i IN 1..l_view_list.count
      LOOP
        EXECUTE IMMEDIATE 'drop view ' || l_view_list(i);
      END LOOP;
    END drop_ecc_desc_views;
  PROCEDURE get_desc_metadata_load_info(
      p_dataset_key   IN VARCHAR2,
      p_dataset_attrs IN ecc_sec_field_values DEFAULT NULL,
      p_languages     IN VARCHAR2 ,
```

```
x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec,
      x_return_status OUT NOCOPY VARCHAR2 )
  IS
    l_apps_schema_name varchar2(25);
    l_desc_sql_text      VARCHAR2(4000)  ;
    l_prec_sql_text      VARCHAR2(4000)  ;
    l_prec_cat1_sql_text VARCHAR2(4000)  := '  SELECT ' ||
'''SHOPPING_CATEGORY_1'' TRIGGER_INSTANCE_ATTRIBUTE, '
    || '''SHOPPING_CATEGORY_2'' TARGET_INSTANCE_ATTRIBUTE, ' || '''*''
TRIGGER_ATTR_VALUE ,' || '''Y''  ENABLED_FLAG '
    || ' FROM dual ';
    l_prec_cat2_sql_text VARCHAR2(4000)  := '  SELECT ' ||
'''SHOPPING_CATEGORY_2'' TRIGGER_INSTANCE_ATTRIBUTE, '
    || '''SHOPPING_CATEGORY_3'' TARGET_INSTANCE_ATTRIBUTE, ' || '''*''
TRIGGER_ATTR_VALUE ,' || '''Y''  ENABLED_FLAG '
    || ' FROM dual ';
    l_prec_cat3_sql_text VARCHAR2(4000)  := '  SELECT ' ||
'''SHOPPING_CATEGORY_3'' TRIGGER_INSTANCE_ATTRIBUTE, '
    || '''SHOPPING_CATEGORY'' TARGET_INSTANCE_ATTRIBUTE, ' || '''*''
TRIGGER_ATTR_VALUE ,' || '''Y''  ENABLED_FLAG '
    || ' FROM dual ';
    query_det_arr ecc_query_det_arr_type := ecc_query_det_arr_type
(NULL);
  BEGIN
    begin
        select oracle_username
        into l_apps_schema_name
        from fnd_oracle_userid
        where read_only_flag = 'U';
    exception
      when others then
        l_apps_schema_name := 'APPS';
    end;

    l_desc_sql_text  := 'SELECT distinct Decode(UPPER(regexp_replace
(key,''[-*# ]'',''_'') ), ''SIZE'', ''SIZE_'', UPPER(regexp_replace
(key,''[-*# ]'',''_'') ))  NAME,
                                ATTRIBUTE_NAME DISPLAY_NAME,
                                Decode(SubStr(stored_in_column,1,3) ,
''NUM'', ''NUMBER'', ''VARCHAR2'') TYPE,
                                NULL SCALE,
                                NULL PRECISION,
                                decode(rt_category_id, 0, ''Y'', ''N'')
SHOW_IN_GUDIED_DISCOVERY,
                                ''REFINEMENTS'' GROUP_KEY,
                                Decode(stored_in_TABLE ,
''PO_ATTRIBUTE_VALUES_TLP'', ''Y'', ''N'') TRANSLATABLE,
                                LANGUAGE
                                FROM icx_cat_attributes_tl
                                WHERE stored_in_table IN
(''PO_ATTRIBUTE_VALUES'', ''PO_ATTRIBUTE_VALUES_TLP'')
                                AND STORED_IN_COLUMN IS NOT NULL
                                AND KEY NOT IN
(''PREFERRED_SUPPLIER_ICON'', ''ECO_FRIENDLY_SUPPLIER_ICON'',
''GREEN_SUPPLIER_ICON'', ''OFF_CONTRACT_ITEM_ICON'',
''OVER_PRICED_ICON'')
                                AND ATTRIBUTE_ID>28
                                AND LANGUAGE IN ( select COLUMN_VALUE
from TABLE(FND_ECC_UTIL_MLS_PVT.GET_LANG_FOR_METADATA(''' || p_languages
|| ''')))';

    l_prec_sql_text  := '  SELECT ' || '''SHOPPING_CATEGORY_1''
TRIGGER_INSTANCE_ATTRIBUTE, '
    || ' UPPER(SubStrB(icx_ecc_util_pvt.makeNCName(ita.KEY), 1,30) )
TARGET_INSTANCE_ATTRIBUTE, '
    || '''*''   TRIGGER_ATTR_VALUE ,' || '''Y''  ENABLED_FLAG '
```

```
|| ' FROM icx_cat_attributes_tl ita, ' || 'fnd_languages lang '
    || ' WHERE lang.language_code= ita.LANGUAGE ' || ' AND lang.
installed_flag =''B'''
    || ' AND   STORED_IN_TABLE   IN
(''PO_ATTRIBUTE_VALUES'',''PO_ATTRIBUTE_VALUES_TLP'') '
    || ' AND STORED_IN_COLUMN like ''%ATTRIBUTE%'''
    || ' AND ita.rt_category_id <> 0 '
    || ' AND upper(icx_ecc_util_pvt.makeNCName(ita.KEY)) in '
    || ' (select column_name FROM  all_tab_columns where table_name like
''ICX_CAT_ECC%VL''  and owner = ''' ||l_apps_schema_name || '''' || '
)'
    || ' GROUP BY ita.KEY';

    icx_ecc_util_pvt.create_ecc_desc_views;
    icx_endeca_util_pkg.seed_icon_descriptors;
    query_det_arr(1) := ecc_query_det_rec(l_desc_sql_text,
icx_ecc_util_pvt.g_icx_exc_att_op);
    query_det_arr.extend ();
    query_det_arr(2) := ecc_query_det_rec(l_prec_sql_text,
icx_ecc_util_pvt.g_icx_exc_pre_op);
    query_det_arr.extend ();
    query_det_arr(3) := ecc_query_det_rec(l_prec_cat1_sql_text,
icx_ecc_util_pvt.g_icx_exc_pre_op);
    query_det_arr.extend ();
    query_det_arr(4) := ecc_query_det_rec(l_prec_cat2_sql_text,
icx_ecc_util_pvt.g_icx_exc_pre_op);
    query_det_arr.extend ();
    query_det_arr(5)  := ecc_query_det_rec(l_prec_cat3_sql_text,
icx_ecc_util_pvt.g_icx_exc_pre_op);
    x_ecc_ds_meta_rec := ecc_ds_meta_rec(p_dataset_key,query_det_arr);
    x_return_status   := 'S';
  END get_desc_metadata_load_info;
  FUNCTION makencname(
      p_attribute_name IN VARCHAR2 )
    RETURN VARCHAR2
  IS
    l_attribute_name VARCHAR2(450);
  BEGIN
    l_attribute_name            := p_attribute_name;
    l_attribute_name            := REPLACE(l_attribute_name,' ','_');
    l_attribute_name            := REPLACE(l_attribute_name,'/','_');
    l_attribute_name            := REPLACE(l_attribute_name,'#','_');
    l_attribute_name            := REPLACE(l_attribute_name,',','_');
    IF ( upper(l_attribute_name) = 'SIZE' ) THEN
      l_attribute_name          := l_attribute_name || '_';
    END IF;
    RETURN l_attribute_name;
  END makencname;

  PROCEDURE SUBMIT_ECC_DATA_LOAD(
              ERRBUF               OUT NOCOPY VARCHAR2 ,
              RETCODE              OUT NOCOPY VARCHAR2 ,
              p_system_name        IN VARCHAR2,
              p_load_type          IN VARCHAR2,
              p_languages          IN VARCHAR2,
              p_log_level          IN VARCHAR2,
              p_trace_enabled      IN VARCHAR2
  ) IS
     l_app_short_name varchar2(10) := 'icx';
     l_request_id   number;
     l_req_data varchar2(10);
     l_req_data1 varchar2(10);
     l_load_type varchar2(200);
  BEGIN

    l_req_data := fnd_conc_global.request_data;
```

```
SELECT decode(p_load_type, 'FULL_LOAD', 'METADATA_LOAD', p_load_type)
INTO l_load_type FROM dual;

    IF l_req_data ='END' THEN
      fnd_file.put_line(FND_FILE.OUTPUT,'Executed the sub request: ICX
ECC Data Load for load type : ' || p_load_type);
      fnd_file.put_line(FND_FILE.LOG,'Executed the sub request: ICX ECC
Data Load for load type : ' || p_load_type);
      RETURN;
    END IF;

    IF (l_req_data ='END-META' and p_load_type = 'FULL_LOAD') THEN
          fnd_file.put_line(FND_FILE.OUTPUT,'Executed the sub request:
ICX ECC Data Load for load type : ' || l_load_type);
          fnd_file.put_line(FND_FILE.LOG,'Executed the sub request: ICX
ECC Data Load for load type : ' || l_load_type);
          l_request_id  := fnd_request.submit_request(
                           application => 'FND',
                           program     => 'ECCRUNDL',
                           description => 'ECC - Run Data Load',
                           start_time  => sysdate,
                           argument1   => p_system_name,
                           argument2   => l_app_short_name,
                           argument3   => null,
                           argument4   => p_load_type,
                           argument5   => p_languages,
                           argument6   => p_trace_enabled,
                           argument7   => p_log_level,
                           sub_request => true);

          fnd_file.put_line(FND_FILE.OUTPUT,'ECC Load for load type : '
|| p_load_type || ' Job Request ID:'||l_request_id);
          fnd_file.put_line(FND_FILE.LOG,'ECC Load for load type : ' ||
p_load_type || ' Job Request ID:'||l_request_id);

          IF l_request_id = 0 THEN
             --
             -- If request submission failed, exit with error.
             --
                ERRBUF := fnd_message.get;
                RETCODE := 2;
          ELSE
             --
             -- Here we set the globals to put the program into the
             -- PAUSED status on exit, and to save the state in
             -- request_data.
             --
             fnd_conc_global.set_req_globals(conc_status => 'PAUSED',
request_data => 'END');

             ERRBUF := 'Sub-Request submitted!';
             RETCODE := 0 ;

          END IF;
          RETURN;
    END IF;
    -- Submit l_load_type which can be either METADATA_LOAD or
INCREMENTAL_LOAD.
    l_request_id  := fnd_request.submit_request(
                          application => 'FND',
                          program     => 'ECCRUNDL',
                          description => 'ECC - Run Data Load',
                          start_time  => sysdate,
                          argument1   => p_system_name,
                          argument2   => l_app_short_name,
                          argument3   => null,
```

```
argument4   => l_load_type,
                         argument5   => p_languages,
                         argument6   => p_trace_enabled,
                         argument7   => p_log_level,
                         sub_request => true);

    fnd_file.put_line(FND_FILE.OUTPUT,'ECC Load for load type : ' ||
l_load_type || ' Job Request ID:'||l_request_id);
    fnd_file.put_line(FND_FILE.LOG,'ECC Load for load type : ' ||
l_load_type || ' Job Request ID:'||l_request_id);

    IF l_request_id = 0 THEN
      --
      -- If request submission failed, exit with error.
      --
      ERRBUF := fnd_message.get;
      RETCODE := 2;
    ELSE
      --
      -- Here we set the globals to put the program into the
      -- PAUSED status on exit, and to save the state in
    -- request_data.
    --
        IF(l_load_type = 'METADATA_LOAD' AND p_load_type='FULL_LOAD')
THEN
          l_req_data1 := 'END-META';
        ELSE
          l_req_data1 := 'END';
        END IF;

    fnd_conc_global.set_req_globals(conc_status => 'PAUSED',request_data
=> l_req_data1);

    ERRBUF := 'Sub-Request submitted!';
    RETCODE := 0 ;

  END IF;
  RETURN;
 EXCEPTION
 WHEN OTHERS THEN
    RETCODE := 1;
    fnd_file.put_line(FND_FILE.OUTPUT,'OTHERS exception while submitting
: ICX ECC Data Load' || sqlerrm);
    fnd_file.put_line(FND_FILE.LOG,'OTHERS exception while submitting :
ICX ECC Data Load' || sqlerrm);

  END SUBMIT_ECC_DATA_LOAD;

END icx_ecc_util_pvt;
/
--show errors;
COMMIT;
EXIT;
```

Example from Order Management

```
CREATE OR REPLACE PACKAGE BODY APPS.OE_ECC_UTIL_PVT
AS
    /* $Header: OEXUEXMP.pls 120.0.1 2019/07/22 06:14:03 noship $ */
    /* Procedure GET_ECC_DATA_LOAD_INFO is the main procedure which is
called during both full load and
    ** incremental load, for both ont-lines as well as ont-headers data
sets.
    **
    ** GET_ECC_DATA_LOAD_INFO - Retrieves the data load details for the
given data set key
    **
    ** IN parameters
    **    p_dataset_id - dataset key
    **    p_load_type - Indicating Full or incremental load
(F/I/Custom).For custom there are no data load rules defined
    **    p_ds_last_success_run - Returns the last successful etl run
for the dataset key
    ** OUT parameters
    **    x_ecc_ds_meta_rec - ecc_ds_meta_rec
    **    x_return_status - return status
    */

--------------------------------------------------------------------------
--------------------------
    PROCEDURE GET_ECC_DATA_LOAD_INFO (
        p_dataset_key           IN            VARCHAR2,
        p_load_type             IN            VARCHAR2,
        p_ds_last_success_run   IN            DATE,
        p_languages             IN            VARCHAR2,
        P_ADDL_PARAMS           IN            ECC_SEC_FIELD_VALUES
DEFAULT NULL,
        x_ecc_ds_meta_rec          OUT NOCOPY ecc_ds_meta_rec,
        x_return_status            OUT NOCOPY VARCHAR2)
    IS
        l_masstrans_d_sql_text       CLOB;
        l_masstrans_i_sql_text       CLOB;
        l_ont_header_full_text       VARCHAR2 (30000);
        l_ont_header_hold_text       VARCHAR2 (2500);
        l_base_language              VARCHAR2 (20);
        l_line_del_sql_text          VARCHAR2 (256)
            := 'select line_id ecc_spec_id from oe_ecc_deletion_map
where line_id is not null';
        l_hdr_del_sql_text           VARCHAR2 (256)
            := 'select distinct header_id ecc_spec_id from
oe_ecc_deletion_map where line_id is null';
        v_for_lang_pivot_clause      VARCHAR2 (400)
            := FND_ECC_UTIL_MLS_PVT.GEN_ECC_MLS_PIVOT_FOR_LANG_CL (
                  p_languages);
        l_ont_line_hold_text         VARCHAR2 (4000)
            :=    'SELECT * FROM
             (SELECT DISTINCT oegt.line_id ecc_spec_id,
               oegt.total_hold_name,
               oegt.header_hold_name,
               oegt.line_hold_name,
               oegt.GRAPH_ALERT_NAME,
               oegt.alert_text,
               oegt.language_code
               FROM oe_ecc_global_temp oegt
               ) pivot (max (GRAPH_ALERT_NAME) as GRAPH_ALERT_NAME,
          max(alert_text) as alert_text
for language_code in ('
               || v_for_lang_pivot_clause
```

```
|| '))';
        l_ont_full_sql_text          VARCHAR2 (8000)
            :=    'SELECT * FROM (SELECT
                                                    ECC_SPEC_ID,
ECC_LAST_UPDATE_DATE,ORDER_NUMBER,LINE_NUMBER,LINE_ID,HEADER_ID,PRODUCT,
LINE_QUANTITY,UNIT_SELLING_PRICE,
                                                    ORDER_UOM,
PRODUCT_DESCRIPTION,LINE_TYPE,PRICE_LIST,PRICE_LIST_ID,ORDER_DATE,
SHIP_FROM_ORG_ID,WAREHOUSE,SOURCE_TYPE,
                                                    SHIPMENT_METHOD,
CARRIER,SHIPMENT_PRIORITY,SHIPPING_INSTRUCTIONS,PACKING_INSTRUCTIONS,
FREIGHT_TERMS,SALESPERSON,CUSTOMER,
                                                    CUSTOMER_NUMBER,
SHIP_TO_CUSTOMER_NAME,SHIP_TO_CUSTOMER_NUMBER,SHIP_TO_CONTACT,
SHIP_TO_CONTACT_ID,SHIP_TO_ADDRESS,SHIP_TO_ORG_ID,

BILL_TO_CUSTOMER_NAME,BILL_TO_CUSTOMER_NUMBER,BILL_TO_CONTACT,
BILL_TO_CONTACT_ID,BILL_TO_ADDRESS,BILL_TO_ORG_ID,EXPECTED_DELAY,
                                                    REQUEST_DATE,
REQUEST_DATE_TYPE,SCHEDULE_SHIP_DATE,ACTUAL_SHIPMENT_DATE,
SCHEDULE_ARRIVAL_DATE,ACTUAL_ARRIVAL_DATE,

LATEST_ACCEPTABLE_DATE,FULFILLMENT_DATE,FLOW_STATUS_CODE,LINE_STATUS,
ITEM_TYPE_CODE,ORG_ID,OPERATING_UNIT,LINE_CATEGORY_CODE,
                                                    FULFILLED_FLAG,
LINE_TRAN_AMOUNT,OPEN_FLAG,TXN_CURRENCY,TXN_CURRENCY_CODE,
FUNC_CURRENCY_CODE,FUNC_CURRENCY,LINE_AMOUNT,
                                                    SHIP_TO_SITE,
SHIP_TO_LOCATION,SHIP_TO_STATE,SHIP_TO_COUNTRY,BILL_TO_SITE,
BILL_TO_CITY,BILL_TO_STATE,BILL_TO_COUNTRY,

LINE_LAST_UPDATE_DATE,PRIMARY_UOM,INVENTORY_ITEM_ID,SALES_CHANNEL,
ORDER_TYPE,STATUS,SHIPPED_QUANTITY,FULFILLED_QUANTITY,
                                                    INVOICED_QUANTITY,
SCHEDULE_STATUS,PROMISE_DATE,CUSTOMER_PO,PAYMENT_TERM,AGREEMENT_NAME,
SALES_AGREEMENT_NUMBER,SALES_AGREEMENT_LINE_NUMBER,
                                                    SUBINVENTORY,
PICK_STATUS,TRANSACTIONAL_ORDER_TOTAL,ORDER_TOTAL,RESERVED_QUANTITY,
PARTY_ID,PREDICTED_RETURN_REASON_CODE,
                                                    ANOMALIES,
LINE_HOLD_NAME,HEADER_HOLD_NAME,TOTAL_HOLD_NAME,ALERT_FLAG,ALERT_COUNT,
ALERT_TEXT,GRAPH_ALERT_NAME,DELAY_FLAG,DELAY_COUNT,
                                                    CONVERSION_RATE,
CONVERSION_TYPE_CODE,RESERVED_QTY,BOOKED_FLAG,ORDER_ALERT_COUNT,
ALERT_TYPE,HDR_ALERT_COUNT,
                                                    LANGUAGE_CODE,
ALERT_TYPE_CODE,TABLE_ALERT_FLAG,HDR_LAST_UPDATE_DATE,ON_ALERT,IS_OPEN,
IS_BOOKED,
                                                    LINE_QUANTITY_1,
SHIPPED_QUANTITY_1,CURRENCY,CSR_USER_NAME
                                                FROM
                                                    OE_ECC_GLOBAL_TEMP )
                                                PIVOT (
                                                    max(freight_terms)
as freight_terms,
                                                    max
(product_description) as product_description,
                                                    max(price_list) as
price_list,
                                                    max(line_type) as
line_type,
                                                    max(payment_term) as
payment_term,
                                                    max(ship_to_contact)
as ship_to_contact,
                                                    max(bill_to_contact)
```

```
as bill_to_contact,
                                                        max(line_status) as
line_status,
                                                        max(operating_unit)
as operating_unit,
                                                        max(sales_channel)
as sales_channel,
                                                        max(SHIPMENT_METHOD)
as SHIPMENT_METHOD,
                                                        max(on_alert) as
on_alert,
                                                        max(is_open) as
is_open,
                                                        max(is_booked) as
is_booked,
                                                        max(order_type) as
order_type,
                                                        max(agreement_name)
as agreement_name,
                                                        max(SOURCE_TYPE) as
SOURCE_TYPE,
                                                        max(TXN_CURRENCY) as
TXN_CURRENCY,
                                                        max(func_currency)
as func_currency,
                                                        max(BILL_TO_COUNTRY)
as BILL_TO_COUNTRY,
                                                        max(ship_to_country)
as ship_to_country,
                                                        max(STATUS) as
STATUS,
                                                        max
(GRAPH_ALERT_NAME) as GRAPH_ALERT_NAME,
          max(alert_text) as alert_text ,
                                                        max(warehouse) as
warehouse,
                                                        max(alert_type) as
alert_type
                                                        for language_code in
('
               || v_for_lang_pivot_clause
               || '))';

        query_det_arr               ecc_query_det_arr_type
                                        := ecc_query_det_arr_type
(NULL);

        CURSOR lines IS SELECT * FROM OE_ECC_GLOBAL_TEMP;
        l_current_header_id          NUMBER := 0;
        l_current_total              NUMBER := 0;
        l_func_total                 NUMBER := 0;
        l_hold_cnt                   NUMBER := 0;
        l_anam_cnt                   NUMBER := 0;
        l_reserved_qty2              NUMBER := 0;
        l_return_status              VARCHAR2 (200);
        l_currency_multiple          VARCHAR2 (200);
        l_exists                     VARCHAR2 (1);
        l_hdr_total_sql_text         VARCHAR2 (2500);
        l_ont_line_dff_text          VARCHAR2 (2500);
        l_ont_header_dff_text        VARCHAR2 (2500);
        l_alert_anomaly              VARCHAR2 (200);
        l_alert_delay                VARCHAR2 (200);
        l_alert_expected_delay       VARCHAR2 (200);
        l_alert_hold                 VARCHAR2 (200);
        l_alert_predictive_alert     VARCHAR2 (200);
        l_alert_pred_return_reason   VARCHAR2 (200);
```

```
l_alert_return_prediction    VARCHAR2 (200);
        l_operating_unit             VARCHAR2 (100);
        l_yes_label_meaning          VARCHAR2 (20);
        l_no_label_meaning           VARCHAR2 (20);
        l_hdr_hold_cnt               NUMBER;
        lines_to_process             SYS_REFCURSOR;
        l_ECC_BULK_LOAD_SIZE         NUMBER
            := NVL (fnd_profile.Value_wnps ('OM_ECC_BULK_LOAD_SIZE'),
10000);
        l_alert_type_tbl             alert_type_tbl;
        l_yes_no_tbl                 alert_type_tbl;
        l_ecc_tbl                    ecc_temp_tbl_type;
        l_full_ecc_tbl               ecc_temp_tbl_type;

        CURSOR alert_types IS
              SELECT meaning, language, lookup_code
                FROM fnd_lookup_values
               WHERE lookup_type = 'ONT_ALERT_TYPE'
            ORDER BY lookup_code;

        CURSOR yes_no IS
              SELECT meaning, language, lookup_code
                FROM fnd_lookup_values
               WHERE     lookup_type = 'YES_NO'
                     AND view_application_id = 660
                     AND enabled_flag = 'Y'
            ORDER BY lookup_code;

        CURSOR request_date_type_cur IS
            SELECT meaning, lookup_code
              FROM oe_lookups
             WHERE lookup_type = 'REQUEST_DATE_TYPE' AND enabled_flag =
'Y';

        TYPE request_date_type_tbl IS TABLE OF request_date_type_cur%
ROWTYPE
            INDEX BY BINARY_INTEGER;

        l_request_date_type_tbl      request_date_type_tbl;
    BEGIN
        -- insert_debug('Inside GET_ECC_DATA_LOAD_INFO procedure');
        -- set_db_trace('enable');
        -- insert_debug('l_ECC_BULK_LOAD_SIZE:'||l_ECC_BULK_LOAD_SIZE);
        -- insert_debug('p_dataset_key:'||p_dataset_key);
        -- insert_debug('p_load_type:'||p_load_type);
        -- insert_debug('p_ds_last_success_run:'||TO_CHAR
(p_ds_last_success_run,'DD-MON-YYYY HH24:MI:SS'));
        -- insert_debug('After truncating oe_ecc_global_temp');
        -- first delete all the records from oe_ecc_deletion_map that
are already processed.
        -- delete from oe_ecc_deletion_map WHERE processed = 'Y';
commented. trying truncation instead.
        -- get the meanings of alert types by opening the cursor

        OPEN alert_types;

        FETCH alert_types BULK COLLECT INTO l_alert_type_tbl;

        CLOSE alert_types;

        OPEN yes_no;

        FETCH yes_no BULK COLLECT INTO l_yes_no_tbl;

        CLOSE yes_no;
```

```
OPEN request_date_type_cur;

        FETCH request_date_type_cur BULK COLLECT INTO
l_request_date_type_tbl;

        CLOSE request_date_type_cur;


        IF p_dataset_key = 'ont-lines'
        THEN
            -- if data set is ont-lines, first truncate all the
necessary temp tables.
            trunc_temp_tables (p_load_type);

            IF (p_load_type = 'FULL_LOAD')
            THEN                                    -- full load of
ont-lines
                OPEN lines_to_process FOR
                      SELECT *
                        FROM (-- Subquery Refactoring
                              WITH
                                  requested_languages
                                  AS
                                      (    SELECT REGEXP_SUBSTR (
                                                  p_languages,
                                                  '[^,]+',
                                                  1,
                                                  LEVEL)    AS
LANGUAGE_CODE
                                             FROM DUAL
                                       CONNECT BY REGEXP_SUBSTR
(p_languages,
                                                                '[^,]
+',
                                                                1,
                                                                LEVEL)
                                                  IS NOT NULL)
                              -- Actual select query
                              SELECT a.*
                                FROM oe_ecc_order_lines_v a,
                                     requested_languages lang
                               WHERE     a.open_flag = 'Y'
                                     AND a.language_code IN
                                            (lang.LANGUAGE_CODE)
                                     AND org_id IS NOT NULL -- added to
avoid count mismatch
                              UNION ALL
                              SELECT /*+ index(OE_ECC_ORDER_LINES_V.l
OE_ORDER_LINES1_ENDECA_N1) */
                                     a.*
                                FROM oe_ecc_order_lines_v a,
                                     requested_languages lang
                               WHERE     a.line_LAST_UPDATE_DATE >=
                                           SYSDATE
                                         - NVL (
                                              FND_PROFILE.VALUE_WNPS (

'OM_ECC_FULL_LOAD_DAYS'),
                                              90)
                                     AND a.open_flag = 'N'
                                     AND a.language_code IN
                                            (lang.LANGUAGE_CODE)
                                     AND a.org_id IS NOT NULL -- added
to avoid count mismatch
                                                              )
                    ORDER BY header_id;
```

```
ELSIF (p_load_type = 'INCREMENTAL_LOAD')
           THEN                                --incremental load of
ont-lines
                OPEN lines_to_process FOR
                    --SELECT  /*+ index(oeol.l
OE_ORDER_LINES1_ENDECA_N1) leading(oeol.l oeol.h) cardinality(oeol.l 10)
*/ *
                    --FROM oe_ecc_order_lines_v oeol
                    --WHERE ( ecc_last_update_date >=
p_ds_last_success_run ) AND language_code in (
                    --       select regexp_substr(p_languages,'[^,]+',
1, level) AS LANGUAGE_CODE from dual
                    --       connect by regexp_substr(p_languages, '[^,]
+', 1, level) is not null)

                    -- Subquery Refactoring
                    WITH
                        requested_languages
                        AS
                            (    SELECT REGEXP_SUBSTR (
                                        p_languages,
                                        '[^,]+',
                                        1,
                                        LEVEL)    AS LANGUAGE_CODE
                                   FROM DUAL
                                 CONNECT BY REGEXP_SUBSTR (p_languages,
                                             '[^,]+',
                                             1,
                                             LEVEL)
                                        IS NOT NULL)
                    -- Actual select query
                    SELECT /*+ index(oeol.h OE_ORDER_LINES6_ENDECA_N1)
leading(oeol.h oeol.l) cardinality(oeol.l 10) */
                           oeol.*
                      FROM oe_ecc_order_lines_v  oeol,
                           requested_languages   lang
                     WHERE     (hdr_last_update_date >=
p_ds_last_success_run)
                           AND oeol.language_code IN (lang.
LANGUAGE_CODE)
                    UNION
                    SELECT /*+ index(oeol.l OE_ORDER_LINES1_ENDECA_N1)
leading(oeol.l oeol.h) cardinality(oeol.l 10) */
                           oeol.*
                      FROM oe_ecc_order_lines_v  oeol,
                           requested_languages   lang
                     WHERE     (line_last_update_date >=
                                 p_ds_last_success_run)
                           AND oeol.language_code IN (lang.
LANGUAGE_CODE)
                    UNION
                    SELECT /*+ leading(oeol.tta oeol.l oeol.h)
cardinality(oeol.l 10) */
                           oeol.*
                      FROM oe_ecc_order_lines_v  oeol,
                           requested_languages   lang
                     WHERE     (line_type_last_update_date >=
                                 p_ds_last_success_run)
                           AND oeol.language_code IN (lang.
LANGUAGE_CODE)
                    UNION
                    SELECT /*+ index(h OE_ORDER_LINES4_ENDECA_N1) */
                           a.*
                      FROM oe_ecc_order_lines_v  a,
                           oe_ordeR_holds_all    h,
                           requested_languages   lang
```

```
WHERE     a.line_id = h.line_id
                          AND h.last_update_date >=
p_ds_last_success_run
                          AND a.language_code IN (lang.LANGUAGE_CODE)
                    UNION
                    SELECT /*+ index(h OE_ORDER_LINES4_ENDECA_N1) */
                          a.*
                      FROM oe_ecc_order_lines_v  a,
                          oe_ordeR_holds_all    h,
                          requested_languages   lang
                     WHERE     a.header_id = h.header_id
                          AND h.last_update_date >=
p_ds_last_success_run
                          AND a.language_code IN (lang.LANGUAGE_CODE)
                    UNION
                    SELECT /*+ index(p OE_PREDICTIONS_N2) */
                          a.*
                      FROM oe_ecc_order_lines_v  a,
                          oe_predictions        p,
                          requested_languages   lang
                     WHERE     a.line_id = p.line_id
                          AND p.last_update_date >=
p_ds_last_success_run
                          AND a.language_code IN (lang.LANGUAGE_CODE);

            END IF;                           -- load type is
incremental load

            LOOP                                      -- fetch
lines_to_process
                FETCH lines_to_process
                    BULK COLLECT INTO l_ecc_tbl
                    LIMIT l_ECC_BULK_LOAD_SIZE;

                EXIT WHEN l_ecc_tbl.COUNT = 0;

                --insert_debug('l_ecc_tbl.COUNT is:'||l_ecc_tbl.COUNT);

                FOR i IN l_ecc_tbl.FIRST .. l_ecc_tbl.LAST
                LOOP
                    l_alert_anomaly := NULL;
                    l_alert_delay := NULL;
                    l_alert_expected_delay := NULL;
                    l_alert_hold := NULL;
                    l_alert_predictive_alert := NULL;
                    l_alert_pred_return_reason := NULL;
                    l_alert_return_prediction := NULL;
                    l_operating_unit := NULL;

                    l_yes_label_meaning := NULL;
                    l_no_label_meaning := NULL;

                    FOR j IN l_alert_type_tbl.FIRST .. l_alert_type_tbl.
LAST
                    LOOP
                        IF l_alert_type_tbl (j).language =
                           l_ecc_tbl (i).language_code
                        THEN
                            IF l_alert_type_tbl (j).lookup_code =
'ANOMALY'
                            THEN
                                l_alert_anomaly :=
                                    l_alert_type_tbl (j).meaning;
                            ELSIF l_alert_type_tbl (j).lookup_code =
'DELAY'
                            THEN
```

```
l_alert_delay := l_alert_type_tbl (j).meaning;
                            ELSIF l_alert_type_tbl (j).lookup_code =
                                  'EXP_DELAY'
                            THEN
                                l_alert_expected_delay :=
                                    l_alert_type_tbl (j).meaning;
                            ELSIF l_alert_type_tbl (j).lookup_code =
'HOLD'
                            THEN
                                l_alert_hold := l_alert_type_tbl (j).
meaning;
                            ELSIF l_alert_type_tbl (j).lookup_code =
                                  'PRED_ALERT'
                            THEN
                                l_alert_predictive_alert :=
                                    l_alert_type_tbl (j).meaning;
                            ELSIF l_alert_type_tbl (j).lookup_code =
                                  'PRE_RET_REASON'
                            THEN
                                l_alert_pred_return_reason :=
                                    l_alert_type_tbl (j).meaning;
                            ELSIF l_alert_type_tbl (j).lookup_code =
                                  'RETURN_PRED'
                            THEN
                                l_alert_return_prediction :=
                                    l_alert_type_tbl (j).meaning;
                            END IF;
                        END IF;
                    END LOOP;


                    FOR j IN l_yes_no_tbl.FIRST .. l_yes_no_tbl.LAST
                    LOOP
                        IF l_yes_no_tbl (j).language =
                           l_ecc_tbl (i).language_code
                        THEN
                            IF l_yes_no_tbl (j).lookup_code = 'Y'
                            THEN
                                l_yes_label_meaning :=
                                    l_yes_no_tbl (j).meaning;
                            ELSIF l_yes_no_tbl (j).lookup_code = 'N'
                            THEN
                                l_no_label_meaning :=
                                    l_yes_no_tbl (j).meaning;
                            END IF;
                        END IF;
                    END LOOP;

                    -- convert line total into functional currency
                    IF l_ecc_tbl (i).FUNC_CURRENCY_CODE IS NULL
                    THEN
                        l_ecc_tbl (i).FUNC_CURRENCY_CODE :=
                            oe_upgrade_misc.get_sob_currency (
                                l_ecc_tbl (i).org_id);

                        IF l_ecc_tbl (i).FUNC_CURRENCY_CODE IS NOT NULL
                        THEN
                            SELECT cur.NAME
                              INTO l_ecc_tbl (i).FUNC_CURRENCY
                              FROM fnd_currencies_tl cur
                             WHERE     cur.currency_code =
                                       l_ecc_tbl (i).FUNC_CURRENCY_CODE
                                   AND cur.LANGUAGE =
                                       l_ecc_tbl (i).LANGUAGE_CODE;
--'US';
                        END IF;
```

```
END IF;

                    IF l_ecc_tbl (i).flow_status_code IS NOT NULL
                    THEN
                        l_ecc_tbl (i).line_status :=
                            oe_ecc_util_pvt.Get_Line_Status (
                                l_ecc_tbl (i).line_id,
                                l_ecc_tbl (i).flow_status_code,
                                l_ecc_tbl (i).language_code);
                    END IF;

                    IF     l_ecc_tbl (i).FUNC_CURRENCY IS NULL
                       AND l_currency_multiple IS NULL
                    THEN
                        BEGIN
                            SELECT MESSAGE_TEXT
                              INTO l_currency_multiple
                              FROM fnd_new_messages
                             WHERE message_name =
                                   'ONT_ENDECA_MULTIPLE_CURRENCIES';
                        EXCEPTION
                            WHEN OTHERS
                            THEN
                                l_currency_multiple := NULL;
                        END;

                        l_ecc_tbl (i).FUNC_CURRENCY :=
l_currency_multiple;
                    ELSIF     l_ecc_tbl (i).FUNC_CURRENCY IS NULL
                          AND l_currency_multiple IS NOT NULL
                    THEN
                        l_ecc_tbl (i).FUNC_CURRENCY :=
l_currency_multiple;
                    END IF;

                    IF NVL (l_ecc_tbl (i).TXN_CURRENCY_CODE, 'abc') <>
                       NVL (l_ecc_tbl (i).FUNC_CURRENCY_CODE, 'abc')
                    THEN
                        OE_UPGRADE_MISC.CONVERT_CURRENCY (
                            l_ecc_tbl (i).line_tran_amount,
                            l_ecc_tbl (i).TXN_CURRENCY_CODE,
                            l_ecc_tbl (i).FUNC_CURRENCY_CODE,
                            l_ecc_tbl (i).order_date,
                            NULL,
                            'Corporate',
                            l_return_status,
                            l_ecc_tbl (i).line_amount);
                    ELSE
                        l_ecc_tbl (i).line_amount :=
                            l_ecc_tbl (i).line_tran_amount;
                    END IF;

                    FOR j IN l_request_date_type_tbl.FIRST ..
                             l_request_date_type_tbl.LAST
                    LOOP
                        IF l_ecc_tbl (i).request_date_type =
                           l_request_date_type_tbl (j).lookup_code
                        THEN
                            l_ecc_tbl (i).request_date_type :=
                                l_request_date_type_tbl (j).meaning;
                        END IF;
                    END LOOP;

                    -- insert_debug('l_ecc_tbl(i).
line_amount:'||l_ecc_tbl(i).line_amount);
                    -- insert_debug
```

```
('l_current_header_id:'||l_current_header_id);
                    -- insert_debug('l_ecc_tbl(i).header_id:'||l_ecc_tbl
(i).header_id);
                    -- insert_debug('l_ecc_tbl(i).
order_total:'||l_func_total);
                    -- insert_debug('l_ecc_tbl(i).
transactional_order_total:'||l_ecc_tbl(i).transactional_order_total);
                    -- derive total holds
                    BEGIN
                          SELECT LISTAGG (holds_tbl.HOLD_NAME, '|')
                                    WITHIN GROUP (ORDER BY holds_tbl.
line_id)    total_hold_name
                            INTO l_ecc_tbl (i).total_hold_name
                            FROM (  SELECT /*+ QB_NAME
(HOLD_APPLY_INFO_SELECT2) */
                                           LISTAGG (hdf.name, '|')
                                               WITHIN GROUP (ORDER BY l.
line_id)
                                               HOLD_NAME,
                                           l.line_id
                                               line_id
                                     FROM oe_order_lines_all l,
                                          oe_order_holds_all h,
                                          oe_hold_sources_all hsr,
                                          oe_hold_definitions hdf,
                                          oe_lookups       oel
                                    WHERE     h.header_id = l.header_id
                                          AND l.line_category_code =
'ORDER'
                                          AND h.released_flag = 'N'
                                          AND h.hold_source_id =
                                              hsr.hold_source_id
                                          AND hdf.hold_id = hsr.hold_id
                                          AND oel.lookup_type =
'HOLD_TYPE'
                                          AND oel.lookup_code = hdf.
type_code
                                          AND l.header_id =
                                              l_ecc_tbl (i).header_id
                                          AND (   h.line_id IS NULL
                                               OR h.line_id =
                                                  l_ecc_tbl (i).
line_id)
                                 GROUP BY l.line_id) holds_tbl
                           WHERE line_id = l_ecc_tbl (i).line_id
                        GROUP BY line_id;
                    --insert_debug('After total holds query:'||l_ecc_tbl
(i).total_hold_name);

                    EXCEPTION
                        WHEN OTHERS
                        THEN
                            --insert_debug('in total holds exception');
                            l_ecc_tbl (i).total_hold_name := NULL;
                    END;

                    -- derive header holds

                    BEGIN
                          SELECT LISTAGG (holds_tbl.HOLD_NAME, '|')
                                    WITHIN GROUP (ORDER BY holds_tbl.
line_id)    header_hold_name
                            INTO l_ecc_tbl (i).header_hold_name
                            FROM (  SELECT LISTAGG (hdf.name, '|')
                                           WITHIN GROUP (ORDER BY lin.
line_id)
```

```
HOLD_NAME,
                                        lin.line_id
                                            line_id
                                   FROM oe_order_headers_all hdr,
                                        oe_order_lines_all lin,
                                        oe_order_holds_all oha,
                                        oe_hold_sources_all hsr,
                                        oe_hold_definitions hdf,
                                        oe_lookups        oel
                                  WHERE     hdr.open_flag = 'Y'
                                        AND oha.hold_source_id =
                                            hsr.hold_source_id
                                        AND hsr.hold_id = hdf.hold_id
                                        AND oha.header_id = hdr.
header_id
                                        AND oha.released_flag = 'N'
                                        AND hdr.header_id = lin.
header_id
                                        AND lin.line_category_code =
'ORDER'
                                        AND oha.line_id IS NULL
                                        AND oel.lookup_type =
'HOLD_TYPE'
                                        AND oel.lookup_code = hdf.
type_code
                                        AND hdr.header_id =
                                            l_ecc_tbl (i).header_id
                                 GROUP BY lin.line_id) holds_tbl
                           WHERE line_id = l_ecc_tbl (i).line_id
                        GROUP BY line_id;
                    --insert_debug('in header holds :'||l_ecc_tbl(i).
header_hold_name);

                    EXCEPTION
                        WHEN OTHERS
                        THEN
                            --insert_debug('in header holds exception');
                            l_ecc_tbl (i).header_hold_name := NULL;
                    END;

                    -- derive line holds

                    BEGIN
                          SELECT /*+ QB_NAME(HOLD_APPLY_INFO_SELECT2) */
                                 LISTAGG (hdf.name, '|')
                                     WITHIN GROUP (ORDER BY l.line_id)
HOLD_NAME
                            INTO l_ecc_tbl (i).line_hold_name
                            FROM oe_order_lines_all l,
                                 oe_order_holds_all h,
                                 oe_hold_sources_all hsr,
                                 oe_hold_definitions hdf,
                                 oe_lookups         oel
                           WHERE     h.header_id = l.header_id
                                 AND l.line_id = h.line_id
                                 AND l.line_category_code = 'ORDER'
                                 AND h.released_flag = 'N'
                                 AND h.hold_source_id = hsr.
hold_source_id
                                 AND hdf.hold_id = hsr.hold_id
                                 AND oel.lookup_type = 'HOLD_TYPE'
                                 AND oel.lookup_code = hdf.type_code
                                 AND l.line_id = l_ecc_tbl (i).line_id
                        GROUP BY l.line_id;
                    --insert_debug('in line holds:'||l_ecc_tbl(i).
line_hold_name);
```

```
                    EXCEPTION
                        WHEN OTHERS
                        THEN
                            --insert_debug('in line holds exception');
                            l_ecc_tbl (i).line_hold_name := NULL;
                    END;

                    -- derive line level anomalies
                    BEGIN
                        SELECT LISTAGG (ol.meaning, '|')
                                    WITHIN GROUP (ORDER BY op.line_id)
anomalies
                          INTO l_ecc_tbl (i).anomalies
                          FROM fnd_lookup_values ol, oe_predictions op
                         WHERE     op.anomaly_code = ol.lookup_code
                               AND ol.lookup_type = 'ONT_ANOMALY_CODES'
                               AND ol.language = l_ecc_tbl (i).
LANGUAGE_CODE --'US'//saadepu
                               AND op.line_id = l_ecc_tbl (i).line_id;
                    EXCEPTION
                        WHEN OTHERS
                        THEN
                            --insert_debug('in Anomalies exception');
                            l_ecc_tbl (i).anomalies := NULL;
                    END;

                    -- derive return predictions
                    BEGIN
                        SELECT ol.meaning       --
PREDICTED_RETURN_REASON_CODE
                          INTO l_ecc_tbl (i).
PREDICTED_RETURN_REASON_CODE
                          FROM fnd_lookup_values ol, oe_predictions op
                         WHERE     op.predicted_return_reason_code =
                                   ol.lookup_code
                               AND ol.lookup_type =
'ONT_PRED_RETURN_REASON'
                               AND ol.language = l_ecc_tbl (i).
LANGUAGE_CODE --'US'//saadepu
                               AND op.line_id = l_ecc_tbl (i).line_id;
                    EXCEPTION
                        WHEN OTHERS
                        THEN
                            --insert_debug('in
PREDICTED_RETURN_REASON_CODE exception');
                            l_ecc_tbl (i).PREDICTED_RETURN_REASON_CODE :
=
                                NULL;
                    END;

                    --insert_debug('l_ecc_tbl(i).LINE_ID:'||l_ecc_tbl
(i).LINE_ID);
                    --insert_debug('l_ecc_tbl(i).
SHIP_FROM_ORG_ID:'||l_ecc_tbl(i).SHIP_FROM_ORG_ID);
                    OE_LINE_UTIL.Get_Reserved_Quantities (
                        p_header_id            => l_ecc_tbl (i).
header_id,
                        p_line_id              => l_ecc_tbl (i).LINE_ID,
                        p_org_id               => l_ecc_tbl (i).
SHIP_FROM_ORG_ID,
                        x_reserved_quantity    => l_ecc_tbl (i).
reserved_qty,
                        x_reserved_quantity2   => l_reserved_qty2);

                    --insert_debug('l_ecc_tbl(i).
reserved_qty:'||l_ecc_tbl(i).reserved_qty);
```

```
--insert_debug('l_ecc_tbl(i).ANOMALIES:'||l_ecc_tbl(i).ANOMALIES);
                    --insert_debug('l_ecc_tbl(i).
PREDICTED_RETURN_REASON_CODE:'||l_ecc_tbl(i).
PREDICTED_RETURN_REASON_CODE);
                    --insert_debug('l_ecc_tbl(i).
total_hold_name:'||l_ecc_tbl(i).total_hold_name);
                    --insert_debug('l_ecc_tbl(i).
header_hold_name:'||l_ecc_tbl(i).header_hold_name);
                    --insert_debug('l_ecc_tbl(i).
line_hold_name:'||l_ecc_tbl(i).line_hold_name);

                    IF l_ecc_tbl (i).expected_delay <= 0
                    THEN -- negative expected delay essentially means no
delay
                        l_ecc_tbl (i).expected_delay := NULL;
                    END IF;

                    IF     (   l_ecc_tbl (i).ANOMALIES IS NOT NULL
                            OR l_ecc_tbl (i).
PREDICTED_RETURN_REASON_CODE
                                  IS NOT NULL
                            OR l_ecc_tbl (i).total_hold_name IS NOT NULL
                            OR (    l_ecc_tbl (i).expected_delay IS NOT
NULL
                                AND l_ecc_tbl (i).expected_delay > 0
                                AND NVL (l_ecc_tbl (i).shipped_quantity,
0) =
                                    0))
                       AND l_ecc_tbl (i).open_flag = 'Y'
                    THEN
                        l_ecc_tbl (i).alert_flag := 'Y';
                        l_ecc_tbl (i).table_alert_flag := 'ecc_warning';
                    ELSE
                        l_ecc_tbl (i).alert_flag := 'N';
                        l_ecc_tbl (i).table_alert_flag := 'ecc_blank';
                    END IF;

                    --insert_debug('l_ecc_tbl(i).alert_flag:'||l_ecc_tbl
(i).alert_flag);

                    l_ecc_tbl (i).alert_count := 0;

                    IF l_ecc_tbl (i).anomalies IS NOT NULL
                    THEN
                        l_anam_cnt := 0;

                        SELECT   REGEXP_COUNT (l_ecc_tbl (i).anomalies,
','')
                               + 1
                          INTO l_anam_cnt
                          FROM DUAL;

                        l_ecc_tbl (i).alert_count :=
                            l_ecc_tbl (i).alert_count + l_anam_cnt;
                        l_ecc_tbl (i).alert_type :=
                               l_ecc_tbl (i).alert_type
                            || l_alert_predictive_alert; --
l_alert_type_tbl(5).meaning;      --'Predictive Alerts';
                        l_ecc_tbl (i).alert_type_code := 'PA';
--'Predictive Alerts';
                        l_ecc_tbl (i).alert_text := l_ecc_tbl (i).
anomalies; --(Anomaly:Quantity Anomaly)
                        l_ecc_tbl (i).graph_alert_name :=
l_alert_anomaly; --l_alert_type_tbl(1).meaning;
--l_ecc_tbl(i).anomalies;
                    END IF;
```

```
                    IF l_ecc_tbl (i).PREDICTED_RETURN_REASON_CODE IS NOT
NULL
                    THEN
                        l_ecc_tbl (i).alert_count :=
                            l_ecc_tbl (i).alert_count + 1;

                        IF l_ecc_tbl (i).alert_type IS NOT NULL
                        THEN
                            l_ecc_tbl (i).alert_type :=
                                   l_ecc_tbl (i).alert_type
                                || '|'
                                || l_alert_predictive_alert; --
l_alert_type_tbl(5).meaning; --Predictive Alerts
                            l_ecc_tbl (i).alert_type_code := 'PA'; --
Predictive Alerts
                        ELSE
                            l_ecc_tbl (i).alert_type :=
                                l_alert_predictive_alert; --
l_alert_type_tbl(5).meaning;
                            l_ecc_tbl (i).alert_type_code := 'PA';
                        END IF;          --l_ecc_tbl(i).alert_type IS
NOT NULL

                        IF l_ecc_tbl (i).alert_text IS NOT NULL
                        THEN
                            l_ecc_tbl (i).alert_text :=
                                   l_ecc_tbl (i).alert_text
                                || '|'
                                || l_alert_pred_return_reason --
l_alert_type_tbl(6).meaning
                                || ': '
                                || l_ecc_tbl (i).
PREDICTED_RETURN_REASON_CODE;
                        ELSE
                            l_ecc_tbl (i).alert_text :=
                                   l_alert_pred_return_reason --
l_alert_type_tbl(6).meaning
                                || ': '
                                || l_ecc_tbl (i).
PREDICTED_RETURN_REASON_CODE;
                        END IF;          --l_ecc_tbl(i).alert_text IS
NOT NULL

                        IF l_ecc_tbl (i).graph_alert_name IS NOT NULL
                        THEN
                            l_ecc_tbl (i).graph_alert_name :=
                                   l_ecc_tbl (i).graph_alert_name
                                || '|'
                                || l_alert_return_prediction; --
l_alert_type_tbl(7).meaning; --Return Prediction
                        ELSE
                            l_ecc_tbl (i).graph_alert_name :=
                                l_alert_return_prediction; --
l_alert_type_tbl(7).meaning; --return prediction--l_ecc_tbl(i).
PREDICTED_RETURN_REASON_CODE;
                        END IF;
                    END IF; --l_ecc_tbl(i).PREDICTED_RETURN_REASON_CODE
IS NOT NULL

                    /*hdr_alert_count stores the number of holds at
header level*/

                    IF l_ecc_tbl (i).header_hold_name IS NOT NULL
                    THEN
                        l_hdr_hold_cnt := 0;
```

```
SELECT   REGEXP_COUNT (
                                     l_ecc_tbl (i).header_hold_name,
                                     '\|')
                            + 1
                      INTO l_hdr_hold_cnt
                      FROM DUAL;

                     l_ecc_tbl (i).hdr_alert_count :=
                           NVL (l_ecc_tbl (i).hdr_alert_count, 0)
                         + l_hdr_hold_cnt;
                  --insert_debug('After hdr_alert_count check
l_hdr_hold_cnt is:'||l_hdr_hold_cnt);
                  --insert_debug('After hdr_alert_count check count
is:'||l_ecc_tbl(i).hdr_alert_count);

                  END IF;       -- l_ecc_tbl(i).header_hold_name IS
NOT NULL

                  IF l_ecc_tbl (i).line_hold_name IS NOT NULL
                  THEN
                     l_hold_cnt := 0;

                     SELECT   REGEXP_COUNT (l_ecc_tbl (i).
line_hold_name,
                                            '\|')
                            + 1
                      INTO l_hold_cnt
                      FROM DUAL;

                     --insert_debug('After line_alert_count check
l_hold_cnt is:'||l_hold_cnt);

                     l_ecc_tbl (i).alert_count :=
                         l_ecc_tbl (i).alert_count + l_hold_cnt;

                     --insert_debug('After line_alert_count check
count is:'||l_ecc_tbl(i).alert_count);

                     IF l_ecc_tbl (i).alert_text IS NOT NULL
                     THEN
                         l_ecc_tbl (i).alert_text :=
                                l_ecc_tbl (i).alert_text
                             || '|'
                             || l_alert_hold  --l_alert_type_tbl(4).
meaning
                             || ': '
                             || l_ecc_tbl (i).line_hold_name;
                     ELSE
                         l_ecc_tbl (i).alert_text :=
                                l_alert_hold  --l_alert_type_tbl(4).
meaning
                             || ': '
                             || l_ecc_tbl (i).line_hold_name;
                     END IF;     --l_ecc_tbl(i).alert_text      IS
NOT NULL
                  END IF;         -- l_ecc_tbl(i).line_hold_name IS
NOT NULL

                  IF l_ecc_tbl (i).total_hold_name IS NOT NULL
                  THEN
                     IF l_ecc_tbl (i).alert_type IS NOT NULL
                     THEN
                         l_ecc_tbl (i).alert_type :=
                                l_ecc_tbl (i).alert_type
                             || '|'
                             || l_alert_hold; --l_alert_type_tbl(4).
```

```
meaning;
                        ELSE
                            l_ecc_tbl (i).alert_type := l_alert_hold; --
l_alert_type_tbl(4).meaning;
                        END IF;         -- l_ecc_tbl(i).alert_type IS
NOT NULL
                    END IF;

                    IF l_ecc_tbl (i).total_hold_name IS NOT NULL
                    THEN
                        IF l_ecc_tbl (i).graph_alert_name IS NOT NULL
                        THEN
                            l_ecc_tbl (i).graph_alert_name :=
                                    l_ecc_tbl (i).graph_alert_name
                                || '|'
                                || l_ecc_tbl (i).total_hold_name;
                        ELSE
                            l_ecc_tbl (i).graph_alert_name :=
                                l_ecc_tbl (i).total_hold_name;
                        END IF;
                    END IF;

                    IF     l_ecc_tbl (i).expected_delay IS NOT NULL
                       AND l_ecc_tbl (i).expected_delay > 0
                       AND NVL (l_ecc_tbl (i).shipped_quantity, 0) = 0
                    THEN
                        l_ecc_tbl (i).alert_count :=
                            l_ecc_tbl (i).alert_count + 1;

                        IF l_ecc_tbl (i).alert_type IS NOT NULL
                        THEN
                            l_ecc_tbl (i).alert_type :=
                                    l_ecc_tbl (i).alert_type
                                || '|'
                                || l_alert_expected_delay; --
l_alert_type_tbl(3).meaning;
                        ELSE
                            l_ecc_tbl (i).alert_type :=
                                l_alert_expected_delay; --
l_alert_type_tbl(3).meaning;
                        END IF;       -- l_ecc_tbl(i).alert_type   IS
NOT NULL

                        IF l_ecc_tbl (i).alert_text IS NOT NULL
                        THEN
                            l_ecc_tbl (i).alert_text :=
                                    l_ecc_tbl (i).alert_text
                                || '|'
                                || l_alert_expected_delay; --
l_alert_type_tbl(3).meaning;
                        ELSE
                            l_ecc_tbl (i).alert_text :=
                                l_alert_expected_delay; --
l_alert_type_tbl(3).meaning;
                        END IF;   -- l_ecc_tbl(i).alert_text      IS NOT
NULL.

                        IF l_ecc_tbl (i).graph_alert_name IS NOT NULL
                        THEN
                            l_ecc_tbl (i).graph_alert_name :=
                                    l_ecc_tbl (i).graph_alert_name
                                || '|'
                                || l_alert_expected_delay; --
l_alert_type_tbl(3).meaning;
                        ELSE
                            l_ecc_tbl (i).graph_alert_name :=
```

```
l_alert_expected_delay; --l_alert_type_tbl(3).meaning;
                        END IF;
                    END IF; -- end if for expected delay is not null
condition

                    --insert_debug('l_ecc_tbl(i).
alert_count:'||l_ecc_tbl(i).alert_count);
                    --insert_debug('l_ecc_tbl(i).
expected_delay:'||l_ecc_tbl(i).expected_delay);

                    IF (    l_ecc_tbl (i).expected_delay IS NOT NULL
                        AND l_ecc_tbl (i).expected_delay > 0
                        AND NVL (l_ecc_tbl (i).shipped_quantity, 0) = 0)
                    THEN
                        l_ecc_tbl (i).delay_flag := 'Y';
                        l_ecc_tbl (i).delay_count := 0;

                        IF (    l_ecc_tbl (i).expected_delay IS NOT NULL
                            AND l_ecc_tbl (i).expected_delay > 0)
                        THEN
                            l_ecc_tbl (i).delay_count :=
                                l_ecc_tbl (i).delay_count + 1;
                        END IF;
                    END IF;

                    IF l_ecc_tbl (i).csr_user_id IS NOT NULL
                    THEN
                        BEGIN
                            SELECT p.full_name
                              INTO l_ecc_tbl (i).CSR_USER_NAME
                              FROM per_all_people_f p, fnd_user u
                             WHERE     p.person_id = u.employee_id
                                   AND TRUNC (SYSDATE) BETWEEN p.
effective_start_date
                                                           AND p.
effective_end_date
                                   AND u.user_id = l_ecc_tbl (i).
csr_user_id;
                        EXCEPTION
                            WHEN OTHERS
                            THEN
                                --insert_debug('exception while fetching
CSR_USER_NAME ');
                                l_ecc_tbl (i).CSR_USER_NAME := NULL;
                        END;
                    END IF;

                    --insert_debug('l_ecc_tbl(i).delay_flag:'||l_ecc_tbl
(i).delay_flag );
                    --insert_debug('l_ecc_tbl(i).
delay_count:'||l_ecc_tbl(i).delay_count);

                    IF (p_load_type = 'INCREMENTAL_LOAD')
                    THEN                      -- incremental load of
ont-lines
                        -- first check if record already exists. if
record already exists, just update the alert count and total
                        -- for that record.

                        /*BEGIN

                          SELECT 'Y'
                          INTO l_exists
                          FROM oe_ecc_header_totals
                          WHERE line_id=l_ecc_tbl(i).line_id;*/
```

```
UPDATE oe_ecc_header_totals
                            SET header_id = l_ecc_tbl (i).header_id,
                                line_id = l_ecc_tbl (i).line_id,
                                alert_count = l_ecc_tbl (i).alert_count,
                                hdr_alert_count =
                                    l_ecc_tbl (i).hdr_alert_count,
                                tran_amount = l_ecc_tbl (i).
line_tran_amount,
                                func_amount = l_ecc_tbl (i).line_amount,
                                language_code = l_ecc_tbl (i).
language_code
                             WHERE     line_id = l_ecc_tbl (i).line_id
                                   AND language_code =
                                       l_ecc_tbl (i).language_code;

                            IF SQL%ROWCOUNT = 0
                            THEN
                                INSERT INTO oe_ecc_header_totals (
                                                HEADER_ID,
                                                LINE_ID,
                                                ALERT_COUNT,
                                                HDR_ALERT_COUNT,
                                                tran_amount,
                                                func_amount,
                                                language_code
                                                             )
                                     VALUES (l_ecc_tbl (i).header_id,
                                             l_ecc_tbl (i).line_id,
                                             l_ecc_tbl (i).alert_count,
                                             l_ecc_tbl (i).hdr_alert_count,
                                             l_ecc_tbl (i).line_tran_amount,
                                             l_ecc_tbl (i).line_amount,
                                             l_ecc_tbl (i).language_code
                                                                        );
                            END IF;

                            DELETE FROM oe_ecc_header_totals
                                  WHERE line_id IN (SELECT line_id
                                                      FROM
oe_ecc_deletion_map
                                                     WHERE line_id IS NOT
NULL);

                            /*EXCEPTION
                            WHEN NO_DATA_FOUND THEN
                              -- line does not exist already.Looks like line
got created.So insert into oe_ecc_header_totals


                            END ;*/

                            -- also mark the header into oe_ecc_headers
table, for headers data set to pick the records and process.

                            IF l_current_header_id <> l_ecc_tbl (i).
header_id
                            THEN
                                -- added the IF condition to minimize the
number of inserts into temp table.
                                INSERT INTO oe_ecc_headers (HEADER_ID)
                                     VALUES (l_ecc_tbl (i).header_id);

                                l_current_header_id := l_ecc_tbl (i).
header_id;
                            END IF;
                        END IF;
```

```
                    -- set the lookup based values for open, booked and
alert flags

                    IF NVL (l_ecc_tbl (i).alert_flag, NULL) = 'Y'
                    THEN
                        l_ecc_tbl (i).on_alert := l_yes_label_meaning;
--l_yes_no_tbl(2).meaning;
                    ELSE
                        l_ecc_tbl (i).on_alert := l_no_label_meaning; --
l_yes_no_tbl(1).meaning;
                    END IF;

                    IF NVL (l_ecc_tbl (i).open_flag, NULL) = 'Y'
                    THEN
                        l_ecc_tbl (i).is_open := l_yes_label_meaning; --
l_yes_no_tbl(2).meaning;
                    ELSE
                        l_ecc_tbl (i).is_open := l_no_label_meaning; --
l_yes_no_tbl(1).meaning;
                    END IF;

                    IF NVL (l_ecc_tbl (i).booked_flag, NULL) = 'Y'
                    THEN
                        l_ecc_tbl (i).is_booked := l_yes_label_meaning;
--l_yes_no_tbl(2).meaning;
                    ELSE
                        l_ecc_tbl (i).is_booked := l_no_label_meaning;
--l_yes_no_tbl(1).meaning;
                    END IF;
                END LOOP; -- inner loop for processing records bulk
collected.

                FORALL idx IN INDICES OF l_ecc_tbl
                    INSERT INTO oe_ecc_global_temp
                         VALUES l_ecc_tbl (idx);

                COMMIT;

                /*if it is full load, insert all the record totals into
oe_ecc_header_totals table */

                IF p_load_type = 'FULL_LOAD'
                THEN                                    -- full load of
ont-lines
                    l_full_ecc_tbl := l_ecc_tbl;

                    FOR j IN l_full_ecc_tbl.FIRST .. l_full_ecc_tbl.LAST
                    LOOP
                        IF l_full_ecc_tbl (j).open_flag = 'N'
                        THEN
                            l_full_ecc_tbl (j).line_amount := 0;
                        END IF;
                    END LOOP;

                    FORALL idx IN INDICES OF l_ecc_tbl
                        INSERT INTO oe_ecc_header_totals
                             VALUES (l_full_ecc_tbl (idx).header_id,
                                     l_full_ecc_tbl (idx).line_id,
                                     l_full_ecc_tbl (idx).alert_count,
                                     l_full_ecc_tbl (idx).
hdr_alert_count,
                                     l_full_ecc_tbl (idx).
line_tran_amount,
                                     l_full_ecc_tbl (idx).line_amount,
                                     l_full_ecc_tbl (idx).language_code
```

```
);

                    DELETE FROM oe_ecc_header_totals
                          WHERE line_id IN (SELECT line_id
                                              FROM oe_ecc_deletion_map
                                             WHERE line_id IS NOT NULL);
                END IF;

                COMMIT;

                --insert_debug('After inserting into temp table');

                l_ecc_tbl.DELETE ();
            --insert_debug('After deleting data from plsql table. Count
is:'||l_ecc_tbl.count);

            END LOOP;                           -- outer loop for bulk
collect.

            SELECT LANGUAGE_CODE
              INTO l_base_language
              FROM FND_LANGUAGES
             WHERE INSTALLED_FLAG = 'B';


            --update order totals at one go
            UPDATE oe_ecc_global_temp oegt
               SET (oegt.transactional_order_total, oegt.order_total) =
                       (SELECT SUM (oeh1.tran_amount), SUM (oeh1.
func_amount)
                          FROM oe_ecc_header_totals oeh1
                         WHERE     oegt.header_id = oeh1.header_id
                               AND oeh1.language_code = l_base_language

);



            query_det_arr.EXTEND (1);
            query_det_arr (1) :=
                ecc_query_det_rec (l_ont_full_sql_text,
G_ONT_ECC_UPS_OP);

            IF (p_load_type = 'INCREMENTAL_LOAD')
            THEN
                query_det_arr.EXTEND (2);
                query_det_arr (2) :=
                    ecc_query_det_rec (l_line_del_sql_text,
G_ONT_ECC_DEL_OP);
            --  query_det_arr.extend(3);
            --  query_det_arr(3) := ecc_query_det_rec
(l_ont_line_hold_text,G_ONT_ECC_RPL_OP);

            END IF;

            x_ecc_ds_meta_rec :=
                ecc_ds_meta_rec (p_dataset_key, query_det_arr);
            x_return_status := 'S';
        ELSIF p_dataset_key = 'ont-headers'
        THEN
            IF p_load_type = 'FULL_LOAD'
            THEN                                    -- full load of ont-
headers
                --insert_debug('Full load for headers data set');

                l_ont_header_full_text :=
```

```
'  SELECT * FROM (
                                                SELECT order_number,
customer,customer_number,ship_to_location, order_date, status,
order_alert_count,
                                                    order_alert_flag,
ecc_spec_id,header_id,header_hold_name,customer_po,request_date,
order_type,
                                                    org_id,
order_currency,language_code,
                                                    (SELECT p.full_name
FROM per_all_people_f p, fnd_user u WHERE p.person_id = u.employee_id
AND
                                                     trunc(sysdate)
between p.effective_start_date and p.effective_end_date
                                                      AND u.user_id =
csr_user_id) csr_user_name
                                                FROM
                                                (
                                                    SELECT oegt.
order_number,
                                                    oegt.customer,
                                                    oegt.
customer_number,
                                                    ship_su.LOCATION
ship_to_location,
                                                    oegt.order_date,
                                                    oegt.status,
                                                    SUM (oegt.
alert_count) + MAX(nvl(oegt.hdr_alert_count,0)) order_alert_count,
                                                    DECODE (SUM (oegt.
alert_count) + MAX(nvl(oegt.hdr_alert_count,0)),
0,''ecc_blank'',''ecc_warning'') order_alert_flag,
                                                    oegt.header_id
ecc_spec_id,
                                                    oegt.header_id,
                                                    oegt.
header_hold_name,
                                                    oh.cust_po_number
customer_po,
                                                    oh.request_date ,
                                                    oegt.order_type,
                                                    oegt.org_id,
                                                    oegt.func_currency
order_currency,
                                                    oh.csr_user_id,
                                                    oegt.language_code
language_code
                                                FROM
                                                    oe_ecc_global_temp
oegt,
                                                    oe_order_headers_all
oh,

hz_cust_site_uses_all ship_su
                                                WHERE
                                                    oh.header_id
=oegt.header_id AND
                                                    oh.ship_to_org_id =
ship_su.site_use_id(+) AND
                                                    oegt.org_id IS NOT
NULL
                                                GROUP BY
                                                    oegt.header_id,
                                                    oegt.order_number,
                                                    oegt.customer,
                                                    oegt.
```

```
customer_number,
                                                    oegt.order_date,
                                                    oegt.status,
                                                    ship_su.LOCATION,
                                                    oegt.
header_hold_name,
                                                    oh.cust_po_number,
                                                    oh.request_date ,
                                                    oegt.
hdr_alert_count,
                                                    oegt.order_type,
                                                    oegt.org_id,
                                                    oegt.func_currency,
                                                    oh.csr_user_id,
                                                    oegt.language_code)
                                            )
                                            PIVOT (
                                                max(STATUS) as STATUS,
                                                max(ORDER_CURRENCY) as
ORDER_CURRENCY,
                                                max(ORDER_TYPE) as
ORDER_TYPE
                                                for language_code in ('
                    || v_for_lang_pivot_clause
                    || '))';

                l_hdr_total_sql_text :=
                    'select
                                        oh.header_id ecc_spec_id,
                                        oe_ecc_util_pvt.get_order_total
(oh.header_id,
                                        oh.transactional_curr_code,
                                        oh.org_id,
                                        fnd_profile.Value
(''OM_ECC_DISPLAY_CURRENCY''),
                                                oh.ordered_date,
                                                null,
                                                ''Corporate''
                                            ) order_total
                                    from
                                        oe_order_headers_all oh
                                    where
                                        oh.header_id in
                                            (select distinct header_id
from oe_ecc_global_temp)';

                query_det_arr.EXTEND (1);
                query_det_arr (1) :=
                    ecc_query_det_rec (l_ont_header_full_text,
                                       G_ONT_ECC_UPS_OP);

                query_det_arr.EXTEND (2);
                query_det_arr (2) :=
                    ecc_query_det_rec (l_hdr_total_sql_text,
                                       G_ONT_ECC_UPS_OP);

                x_ecc_ds_meta_rec :=
                    ecc_ds_meta_rec (p_dataset_key, query_det_arr);
                x_return_status := 'S';
            ELSIF (p_load_type = 'INCREMENTAL_LOAD')
            THEN                                --incremental load of ont-
headers
                l_ont_header_full_text :=
                       'SELECT * FROM (
                                            SELECT order_number,
customer,customer_number,ship_to_location, order_date, status,
```

```
order_alert_count,
                                                    order_alert_flag,
ecc_spec_id,header_id,header_hold_name,customer_po,request_date,
order_type,
                                                    org_id,
order_currency,language_code,
                                                    (SELECT p.full_name
FROM per_all_people_f p, fnd_user u WHERE p.person_id = u.employee_id
AND
                                                     trunc(sysdate)
between p.effective_start_date and p.effective_end_date
                                                     AND u.user_id =
csr_user_id) csr_user_name
                                            FROM
                                            (
                                                SELECT  oegt.
order_number ,
                                                    oegt.customer,
                                                    oegt.
customer_number,
                                                    ship_su.LOCATION
ship_to_location,
                                                    oegt.order_date,
oegt.status,
                                                    SUM (oeht.
alert_count) + MAX(NVL(oeht.hdr_alert_count,0)) order_alert_count,
                                                    DECODE (SUM (oegt.
alert_count) + MAX(NVL(oeht.hdr_alert_count,0)),
0,''ecc_blank'',''ecc_warning'') order_alert_flag,
                                                    oegt.header_id
ecc_spec_id,
                                                    oegt.header_id,
                                                    oegt.
header_hold_name,
                                                    oh.cust_po_number
customer_po,
                                                    oh.request_date,
                                                    oegt.order_type,
                                                    oegt.org_id,
                                                    oegt.func_currency
order_currency,
                                                    oh.csr_user_id,
                                                    oegt.language_code
                                                FROM
                                                    oe_ecc_global_temp
oegt,
                                                    oe_ecc_header_totals
oeht,
                                                    oe_order_headers_all
oh,

hz_cust_site_uses_all ship_su
                                                WHERE
                                                    oegt.line_id =oeht.
line_id AND
                                                    oegt.header_id =oh.
header_id AND
                                                    oeht.header_id =oh.
header_id AND
                                                    oh.ship_to_org_id =
ship_su.site_use_id(+) AND
                                                    oegt.header_id IN
                                                        (SELECT
header_id FROM oe_ecc_headers
                                                        UNION
                                                        SELECT DISTINCT
```

```
header_id FROM oe_ecc_deletion_map )
                                                GROUP BY
                                                    oegt.order_number ,
                                                    oegt.customer,
                                                    oegt.
customer_number,
                                                    ship_su.LOCATION,
                                                    oegt.order_date,
                                                    oegt.status,
                                                    oegt.header_id,
                                                    oegt.
header_hold_name,
                                                    oh.cust_po_number,
                                                    oh.request_date,
                                                    oeht.
hdr_alert_count,
                                                    oegt.order_type,
                                                    oegt.org_id,
                                                    oegt.func_currency,
                                                    oh.csr_user_id,
                                                    oegt.language_code)
                                                )
                                                pivot(
                                                    max(status) as
status,
                                                    max(order_type) as
order_type,
                                                    max(order_currency)
as order_currency
                                                    for language_code in
('
                    || v_for_lang_pivot_clause
                    || '))';


                l_hdr_total_sql_text :=
                    'select  oh.header_id ecc_spec_id,
                                            oe_ecc_util_pvt.
get_order_total(oh.header_id,
oh.transactional_curr_code,
oh.org_id,
fnd_profile.Value(''OM_ECC_DISPLAY_CURRENCY''),
oh.ordered_date,
null,
''Corporate''
) order_total
                                        from
                                            oe_order_headers_all oh
                                        where
                                            oh.header_id in (select
header_id from oe_ecc_headers
                                                                UNION
                                                                SELECT DISTINCT
header_id FROM oe_ecc_deletion_map)';

                l_ont_header_hold_text :=
                    'SELECT DISTINCT oegt.header_id ecc_spec_id, oegt.
header_hold_name FROM oe_ecc_global_temp oegt';

                query_det_arr.EXTEND (1);
                query_det_arr (1) :=
                    ecc_query_det_rec (l_ont_header_full_text,
                                       G_ONT_ECC_UPS_OP);

                query_det_arr.EXTEND (2);
                query_det_arr (2) :=
```

```
ecc_query_det_rec (l_hdr_total_sql_text,
                                    G_ONT_ECC_UPS_OP);

                query_det_arr.EXTEND (3);
                query_det_arr (3) :=
                    ecc_query_det_rec (l_hdr_del_sql_text,
G_ONT_ECC_DEL_OP);

                query_det_arr.EXTEND (4);
                query_det_arr (4) :=
                    ecc_query_det_rec (l_ont_header_hold_text,
                                       G_ONT_ECC_RPL_OP);
                x_ecc_ds_meta_rec :=
                    ecc_ds_meta_rec (p_dataset_key, query_det_arr);

                UPDATE oe_ecc_deletion_map
                   SET processed = 'Y';

                --trunc_temp_tables('END_LOAD');
                x_return_status := 'S';
            END IF;                                     -- load type =
full load
        END IF;                         -- data set is ont-lines or ont-
headers
    --  set_db_trace('disable');
    END GET_ECC_DATA_LOAD_INFO;


    PROCEDURE OM_ECC_DATALOAD (ERRBUF            OUT NOCOPY VARCHAR2,
                               RETCODE           OUT NOCOPY VARCHAR2,
                               SYSTEMNAME     IN            VARCHAR2,
                               LOADTYPE       IN            VARCHAR2,
                               LANGUAGES      IN            VARCHAR2,
                               TRACEENABLED   IN            VARCHAR2,
                               LOGLEVEL       IN            VARCHAR2)
    IS
        l_req_data            VARCHAR2 (1000);
        l_request_id          NUMBER;
        lc_phase              VARCHAR2 (50);
        lc_status             VARCHAR2 (50);
        lc_dev_phase          VARCHAR2 (50);
        lc_dev_status         VARCHAR2 (50);
        lc_message            VARCHAR2 (50);
        l_req_return_status   BOOLEAN;
    BEGIN
        l_req_data := fnd_conc_global.request_data;

        IF l_req_data = 'END'
        THEN
            fnd_file.put_line (FND_FILE.OUTPUT,
                               'Executed the sub request: OM ECC Data
Load');
            fnd_file.put_line (FND_FILE.LOG,
                               'Executed the sub request: OM ECC Data
Load');
            RETURN;
        END IF;

        fnd_file.put_line (
            FND_FILE.LOG,
               'Submitting request for ont for ont-lines and loadType: '
            || loadType);

        l_request_id :=
            FND_REQUEST.SUBMIT_REQUEST (
                application   => 'FND',
```

```
program       => 'ECCRUNDL',
                 description   => 'ECC - Run Data Load',
                 start_time    => SYSDATE,
                 ARGUMENT1     => systemName,
                 ARGUMENT2     => 'ont',
                 ARGUMENT3     => 'ont-lines',
                 ARGUMENT4     => loadType,
                 ARGUMENT5     => LANGUAGES,
                 ARGUMENT6     => TRACEENABLED,
                 ARGUMENT7     => LOGLEVEL,
                 sub_request   => FALSE);

        COMMIT;

        IF l_request_id > 0
        THEN
            fnd_file.put_line (FND_FILE.LOG,
                               'Submitted request id: ' ||
l_request_id);
            fnd_file.put_line (FND_FILE.LOG,
                               'Waitting for submitted request
complete+');

            l_req_return_status :=
                fnd_concurrent.wait_for_request (
                    request_id   => l_request_id,
                    interval     => 3,
                    max_wait     => 0,
                    phase        => lc_phase,
                    STATUS       => lc_STATUS,
                    dev_phase    => lc_dev_phase,
                    dev_status   => lc_dev_status,
                    MESSAGE      => lc_message);


            fnd_file.put_line (FND_FILE.LOG,
                               'Waitting for submitted request
complete-');
            fnd_file.put_line (FND_FILE.LOG,
                               ' lc_dev_phase:' || lc_dev_phase);
            fnd_file.put_line (FND_FILE.LOG,
                               ' lc_dev_status:' || lc_dev_status);
        END IF;

        IF UPPER (lc_dev_phase) = 'COMPLETE'
        THEN
            IF    UPPER (lc_dev_status) = 'NORMAL'
               OR UPPER (lc_dev_status) = 'WARNING'
            THEN
                fnd_file.put_line (
                    FND_FILE.LOG,
                    'Submitting request -> ECCRUNDL: ECC - Run Data Load
for ont-headers');

                BEGIN
                    l_request_id :=
                        FND_REQUEST.SUBMIT_REQUEST (
                            application   => 'FND',
                            program       => 'ECCRUNDL',
                            description   => 'ECC - Run Data Load',
                            start_time    => SYSDATE,
                            ARGUMENT1     => systemName,
                            ARGUMENT2     => 'ont',
                            ARGUMENT3     => 'ont-headers',
                            ARGUMENT4     => loadType,
                            ARGUMENT5     => LANGUAGES,
```

```
ARGUMENT6     => TRACEENABLED,
                          ARGUMENT7     => LOGLEVEL,
                          sub_request   => FALSE);
                      COMMIT;
                      fnd_file.put_line (
                          FND_FILE.OUTPUT,
                          'ECC Load Job Request ID:' || l_request_id);
                      fnd_file.put_line (
                          FND_FILE.LOG,
                          'ECC Load Job Request ID:' || l_request_id);
                  EXCEPTION
                      WHEN OTHERS
                      THEN
                          fnd_file.put_line (
                              FND_FILE.LOG,
                                 'OTHERS exception while submitting 2: '
                              || SQLERRM);
                  END;
              END IF;

              IF l_request_id > 0
              THEN
                  fnd_file.put_line (FND_FILE.LOG,
                                     'Submitted request id: ' ||
l_request_id);
                  fnd_file.put_line (
                      FND_FILE.LOG,
                      'Waitting for submitted request complete+');

                  l_req_return_status :=
                      fnd_concurrent.wait_for_request (
                          request_id   => l_request_id,
                          interval     => 3,
                          max_wait     => 0,
                          phase        => lc_phase,
                          STATUS       => lc_STATUS,
                          dev_phase    => lc_dev_phase,
                          dev_status   => lc_dev_status,
                          MESSAGE      => lc_message);


                  fnd_file.put_line (
                      FND_FILE.LOG,
                      'Waitting for submitted request complete-');
                  fnd_file.put_line (FND_FILE.LOG,
                                     ' lc_dev_phase:' || lc_dev_phase);
                  fnd_file.put_line (FND_FILE.LOG,
                                     ' lc_dev_status:' || lc_dev_status);
              END IF;
          END IF;

          --fnd_conc_global.set_req_globals(conc_status => 'PAUSED',
request_data => 'END');
          IF lc_dev_phase = 'COMPLETE'
          THEN
              IF lc_dev_status = 'WARNING'
              THEN
                  RETCODE := 1;
              ELSIF lc_dev_status = 'ERROR'
              THEN
                  RETCODE := 2;
              ELSE
                  RETCODE := 0;
              END IF;
          END IF;
```

```
ERRBUF := 'Sub-Request submitted!';
    EXCEPTION
        WHEN OTHERS
        THEN
            RETCODE := 1;
            fnd_file.put_line (
                FND_FILE.OUTPUT,
                   'OTHERS exception while submitting : WSH ECC Data
Load'
                || SQLERRM);
            fnd_file.put_line (
                FND_FILE.LOG,
                   'OTHERS exception while submitting : WSH ECC Data
Load'
                || SQLERRM);
    END OM_ECC_DATALOAD;


    PROCEDURE get_desc_metadata_load_info (
        p_dataset_key       IN            VARCHAR2,
        p_dataset_attrs     IN            ecc_sec_field_values DEFAULT
NULL,
        p_languages         IN            VARCHAR2,
        x_ecc_ds_meta_rec      OUT NOCOPY ecc_ds_meta_rec,
        x_return_status        OUT NOCOPY VARCHAR2)
    IS
        --l_ecc_dff_segments varchar2(40) := fnd_profile.value('OM: ECC
DFF SEGMENTS');
        query_det_arr     ecc_query_det_arr_type
                              := ecc_query_det_arr_type (NULL);

        l_view_stmt       VARCHAR2 (4000);
        l_column_prefix   VARCHAR2 (2500);
    --l_dff_query_tbl         FND_ECC_DFF_UTIL.dff_query_tbl;
    --vrow                    FND_ECC_DFF_UTIL.dff_query_rec;

    BEGIN
        --insert_debug('Inside get_desc_metadata_load_info' ||
p_dataset_key ||  ' - '  || p_languages);
        /*
        vrow.flexfield_name         := 'OE_HEADER_ATTRIBUTES';
        vrow.app_id                 := 660;
        vrow.column_prefix          := 'ORDER_';
        vrow.attribute_metadata     := FND_ECC_DFF_UTIL.
FND_ECC_DFF_ATTRIBUTES;
        vrow.precedence_metadata    := FND_ECC_DFF_UTIL.
FND_ECC_DFF_PRECEDENCE;
        vrow.context_metadata       := FND_ECC_DFF_UTIL.
FND_ECC_DFF_CONTEXT;

        l_dff_query_tbl(l_dff_query_tbl.count) :=vrow;

        vrow.flexfield_name         := 'OE_LINE_ATTRIBUTES';
        vrow.app_id                 := 660;
        vrow.column_prefix          := '';
        vrow.attribute_metadata     := FND_ECC_DFF_UTIL.
FND_ECC_DFF_ATTRIBUTES;
        vrow.precedence_metadata    := FND_ECC_DFF_UTIL.
FND_ECC_DFF_PRECEDENCE;
        vrow.context_metadata       := FND_ECC_DFF_UTIL.
FND_ECC_DFF_CONTEXT;

        l_dff_query_tbl(l_dff_query_tbl.count) :=vrow;

        FND_ECC_DFF_UTIL.GET_ECC_DFF_INFO(p_dataset_key,p_languages,
l_dff_query_tbl, x_ecc_ds_meta_rec ,x_return_status);*/
```

```
        x_return_status := 'S';
    END get_desc_metadata_load_info;


    /*  This function is copied from   OE_LINE_STATUS_PUB.
Get_Line_Status
     *  Modified to add language as input parameter
     */
    FUNCTION Get_Line_Status (p_line_id            IN NUMBER,
                              p_flow_status_code   IN VARCHAR2,
                              p_language_code      IN VARCHAR2)
        RETURN VARCHAR2
    IS
        l_flow_status_code       VARCHAR2 (80);
        l_flow_meaning           VARCHAR2 (80);
        released_count           NUMBER;
        total_count              NUMBER;
        l_debug_level   CONSTANT NUMBER := oe_debug_pub.g_debug_level;
        l_language_code          VARCHAR2 (10);
    BEGIN
        l_flow_status_code := p_flow_status_code;
        l_language_code := p_language_code;

        IF p_flow_status_code IS NULL
        THEN
            SELECT flow_status_code
              INTO l_flow_status_code
              FROM oe_order_lines
             WHERE line_id = p_line_id;
        END IF;

        IF     l_flow_status_code <> 'AWAITING_SHIPPING'
           AND l_flow_status_code <> 'PRODUCTION_COMPLETE'
           AND l_flow_status_code <> 'PICKED'
           AND l_flow_status_code <> 'PICKED_PARTIAL'
           AND l_flow_status_code <> 'PO_RECEIVED'
           AND l_flow_status_code <> 'SUPPLY_PARTIAL'
        THEN
            SELECT meaning
              INTO l_flow_meaning
              FROM fnd_lookup_values lv
             WHERE     lookup_type = 'LINE_FLOW_STATUS'
                   AND lookup_code = l_flow_status_code
                   AND LANGUAGE = l_language_code
                   AND VIEW_APPLICATION_ID = 660
                   AND SECURITY_GROUP_ID =
                       fnd_global.Lookup_Security_Group (
                           lv.lookup_type,
                           lv.view_application_id);
        ELSE
            SELECT SUM (DECODE (released_status,  'Y', 1,  'C', 1,  0)),
                   SUM (1)
              INTO released_count, total_count
              FROM wsh_delivery_details
             WHERE     source_line_id = p_line_id
                   AND source_code = 'OE'
                   AND released_status <> 'D';

            IF released_count = total_count
            THEN
                SELECT meaning
                  INTO l_flow_meaning
                  FROM fnd_lookup_values lv
                 WHERE     lookup_type = 'LINE_FLOW_STATUS'
                       AND lookup_code = 'PICKED'
                       AND LANGUAGE = l_language_code
```

```
AND VIEW_APPLICATION_ID = 660
                        AND SECURITY_GROUP_ID =
                            fnd_global.Lookup_Security_Group (
                                lv.lookup_type,
                                lv.view_application_id);
                ELSIF released_count < total_count AND released_count <> 0
                THEN
                    SELECT meaning
                      INTO l_flow_meaning
                      FROM fnd_lookup_values lv
                     WHERE     lookup_type = 'LINE_FLOW_STATUS'
                           AND lookup_code = 'PICKED_PARTIAL'
                           AND LANGUAGE = l_language_code
                           AND VIEW_APPLICATION_ID = 660
                           AND SECURITY_GROUP_ID =
                               fnd_global.Lookup_Security_Group (
                                   lv.lookup_type,
                                   lv.view_application_id);
                ELSE
                    SELECT meaning
                      INTO l_flow_meaning
                      FROM fnd_lookup_values lv
                     WHERE     lookup_type = 'LINE_FLOW_STATUS'
                           AND lookup_code = l_flow_status_code
                           AND LANGUAGE = l_language_code
                           AND VIEW_APPLICATION_ID = 660
                           AND SECURITY_GROUP_ID =
                               fnd_global.Lookup_Security_Group (
                                   lv.lookup_type,
                                   lv.view_application_id);
                END IF;
            END IF;

            IF l_debug_level > 0
            THEN
                fnd_file.put_line (FND_FILE.LOG,
                                   'Exiting Get_Line_Status:' ||
l_flow_meaning);
            END IF;

            RETURN l_flow_meaning;
        EXCEPTION
            WHEN NO_DATA_FOUND
            THEN
                NULL;
            WHEN TOO_MANY_ROWS
            THEN
                NULL;
            WHEN OTHERS
            THEN
                NULL;
        END Get_Line_Status;

        PROCEDURE set_db_trace (p_action VARCHAR2)
        IS
        BEGIN
            --insert_debug('Inside enable_db_trace procedure');

            IF p_action = 'enable'
            THEN
                --insert_debug('before enabling trace');
                EXECUTE IMMEDIATE 'ALTER SYSTEM SET max_dump_file_size =
unlimited';

                EXECUTE IMMEDIATE 'ALTER SYSTEM SET timed_statistics =
true';
```

```
            EXECUTE IMMEDIATE 'ALTER session set tracefile_identifier=
''suneela_ecc''';

            EXECUTE IMMEDIATE 'ALTER session set events ''10046  trace
name context forever, level 8''';

            EXECUTE IMMEDIATE 'ALTER session set statistics_level =
''ALL''';
        --insert_debug('after enabling trace');

        ELSIF p_action = 'disable'
        THEN
            --insert_debug('before disabling trace');
            DBMS_SESSION.reset_package;

            EXECUTE IMMEDIATE 'ALTER SESSION SET EVENTS ''10046 trace
name context off''';
        --insert_debug('after disabling trace');

        END IF;
    EXCEPTION
        WHEN OTHERS
        THEN
            NULL;
    --insert_debug(sqlerrm);
    END set_db_trace;

    PROCEDURE get_accessible_orgs (
        p_user_id            IN            NUMBER,
        p_resp_id            IN            NUMBER,
        x_accessible_outab      OUT NOCOPY icx_tbl_varchar240)
    IS
        l_ou_tab           mo_global.OrgIdTab;
        l_tab              icx_tbl_varchar240;
        l_count            NUMBER := 0;
        l_application_id   NUMBER := NULL;
    BEGIN
        BEGIN
            SELECT application_id
              INTO l_application_id
              FROM fnd_responsibility
             WHERE responsibility_id = p_resp_id;
        EXCEPTION
            WHEN OTHERS
            THEN
                l_application_id := 660;
        END;


        Fnd_Global.Apps_Initialize (p_user_id, p_resp_id,
l_application_id);

        mo_global.init ('ONT');

        l_tab := icx_tbl_varchar240 ();

        SELECT TO_CHAR (organization_id)
          BULK COLLECT INTO l_tab
          FROM mo_glob_org_access_tmp;

        x_accessible_outab := l_tab;
    END get_accessible_orgs;

    /*Function get_order_total is used to calculate the order total
during full load and incremental load. The order
    total value is used in ont-headers data set. */
```

```
    FUNCTION get_order_total (p_header_id              NUMBER,
                              p_header_currency_code   VARCHAR2,
                              p_header_org_id          NUMBER,
                              p_func_currency_code     VARCHAR2,
                              p_order_date             DATE,
                              p_conversion_rate        NUMBER,
                              p_conversion_type_code   VARCHAR2)
        RETURN NUMBER
    IS
        l_total_transactional_amt   NUMBER;
        l_order_total               NUMBER;
        l_return_status             VARCHAR2 (100);
        l_func_currency_code        VARCHAR2 (100);
    BEGIN
        -- first select the amount in ordered currency.
        --insert_debug('Inside get_order_total');
        BEGIN
            SELECT SUM (
                       ROUND (
                           NVL (l.ordered_quantity * l.
unit_selling_price, 0),
                           2))
              INTO l_total_transactional_amt
              FROM oe_order_lines_all l
             WHERE l.header_id = p_header_id;
        --insert_debug('l_total_transactional_amt is
:'||l_total_transactional_amt);
        EXCEPTION
            WHEN OTHERS
            THEN
                l_total_transactional_amt := 0;
        END;

        -- convert the ordered amount into currency based on the value
of profile option "ECC display currency"
        --insert_debug('p_func_currency_code:'||p_func_currency_code);
        BEGIN
            IF p_func_currency_code IS NOT NULL
            THEN
                l_func_currency_code := p_func_currency_code;
            ELSE
                l_func_currency_code :=
                    oe_upgrade_misc.get_sob_currency (p_header_org_id);
            --insert_debug
('l_func_currency_code:'||l_func_currency_code);
            END IF;

            --insert_debug
('l_func_currency_code:'||l_func_currency_code);
            IF NVL (l_func_currency_code, 'abc') <>
p_header_currency_code
            THEN
                OE_UPGRADE_MISC.CONVERT_CURRENCY
(l_total_transactional_amt,

p_header_currency_code,
                                                l_func_currency_code,
                                                p_order_date,
                                                NULL,
                                                'Corporate',
                                                l_return_status,
                                                l_order_total);
            --insert_debug('l_order_total is:'||l_order_total);
            ELSE                        -- currency is same. No need to
convert.
                l_order_total := l_total_transactional_amt;
```

```
--insert_debug('in else block l_order_total is:'||l_order_total);
            END IF;
        EXCEPTION
            WHEN OTHERS
            THEN
                RETURN NULL;
        END;

        --insert_debug('before returning l_order_total
is:'||l_order_total);
        RETURN l_order_total;
    END get_order_total;

    /* Procedure trunc_temp_tables is used to truncate the tables before
full load and incremental load.
    OE_ECC_HEADER_TOTALS table needs to be truncated only before full
load. However, the tables OE_ECC_GLOBAL_TEMP
    and OE_ECC_HEADERS need to be truncated before every run, both full
load and incremental load of ont-lines
    data set only. temp tables are not truncated during ont-headers run.
Becuase ont-headers data set works on
    data populated into temp tables via loads on ont-lines data sets.*/

    PROCEDURE trunc_temp_tables (p_load_type IN VARCHAR2)
    IS
        l_api_success   BOOLEAN := FALSE;
        l_status        VARCHAR2 (128);
        l_industry      VARCHAR2 (128);
        l_schema        VARCHAR2 (128);
        l_stmt          VARCHAR2 (256);
    BEGIN
        l_api_success :=
            fnd_installation.get_app_info (
                application_short_name   => 'ONT',
                status                   => l_status,
                industry                 => l_industry,
                oracle_schema            => l_schema);

        IF l_api_success
        THEN
            NULL;
        --insert_debug('api is success');
        ELSE
            NULL;
        --insert_debug('api is failure');
        END IF;

        --l_stmt := 'truncate table test_debug';
        --EXECUTE IMMEDIATE l_stmt;
        --insert_debug('inside trunc_temp_tables');

        IF l_api_success
        THEN                                                    --if
api_success
            l_stmt :=
                'TRUNCATE TABLE ' || l_schema || '.' ||
'OE_ECC_GLOBAL_TEMP';

            --insert_debug('before truncating OE_ECC_GLOBAL_TEMP');
            EXECUTE IMMEDIATE l_stmt;

            l_stmt :=
                'TRUNCATE TABLE ' || l_schema || '.' ||
'OE_ECC_HEADERS';

            --insert_debug('before truncating OE_ECC_HEADERS');
```

```
EXECUTE IMMEDIATE l_stmt;

            IF p_load_type = 'FULL_LOAD'
            THEN
                l_stmt :=
                       'TRUNCATE TABLE '
                    || l_schema
                    || '.'
                    || 'OE_ECC_HEADER_TOTALS';

                --insert_debug('before truncating
OE_ECC_HEADER_TOTALS');
                EXECUTE IMMEDIATE l_stmt;

                l_stmt :=
                       'TRUNCATE TABLE '
                    || l_schema
                    || '.'
                    || 'OE_ECC_DELETION_MAP';

                EXECUTE IMMEDIATE l_stmt;
            END IF;                                    -- load type is full
load.
        END IF;                                                 --if
api_success

        --insert_debug('before truncating OE_ECC_DELETION_MAP');
        --insert_debug('l_stmt:'||l_stmt);
        DELETE FROM oe_ecc_deletion_map
              WHERE processed = 'Y';
    --insert_debug('After truncating');


    END trunc_temp_tables;

    /* ECC V3 ER start */


    FUNCTION get_csr_details (p_sold_to_org_id      NUMBER,
                              p_inventory_item_id   NUMBER)
        RETURN NUMBER
    IS
        l_csr_user      VARCHAR2 (200);
        l_csr_user_id   NUMBER := NULL;
    BEGIN
        IF p_sold_to_org_id IS NOT NULL AND p_inventory_item_id IS NOT
NULL
        THEN
            --insert_debug('both ct and product are present.try to
derive csr');
            fnd_file.put_line (
                FND_FILE.LOG,
                'both ct and product are present.try to derive csr');

            BEGIN
                SELECT csr_user_id
                  INTO l_csr_user_id
                  FROM (  SELECT *
                            FROM oe_csr_assignment_rules
                           WHERE     sold_to_org_id = p_sold_to_org_id
                                 AND (   item_category IN
                                             (SELECT DISTINCT
                                                     miv.
category_concat_segs
                                                FROM
mtl_item_categories_v miv
```

```
WHERE miv.inventory_item_id =

p_inventory_item_id)
                                       OR item_category IS NULL)
                         ORDER BY sold_to_org_id, item_category) a
                  WHERE ROWNUM = 1;


                RETURN l_csr_user_id;
            EXCEPTION
                WHEN NO_DATA_FOUND
                THEN
                    --insert_debug('CSR rule does not exist. Return
null');
                    fnd_file.put_line (
                        FND_FILE.LOG,
                        'CSR rule does not exist. Return null');
                    RETURN NULL;
            END;
        ELSIF p_inventory_item_id IS NULL
        THEN
            --insert_debug('Item details are not present.Return null');
            fnd_file.put_line (FND_FILE.LOG,
                               'Item details are not present.Return
null');
            RETURN NULL;
        ELSIF p_sold_to_org_id IS NULL
        THEN
            --insert_debug('customer is not yet entered on the order.
Cannot assign CSR');
            fnd_file.put_line (
                FND_FILE.LOG,
                'customer is not yet entered on the order. Cannot assign
CSR');
            RETURN NULL;
        END IF;
    EXCEPTION
        WHEN OTHERS
        THEN
            --insert_debug('others exception:'||sqlerrm);
            fnd_file.put_line (FND_FILE.LOG, 'others exception:' ||
SQLERRM);
            RETURN NULL;
    END;


    /* Procedure reassign_csr is called when Assign Customer Servie Rep
CP is invoked.
      **
      **
      ** reassign_csr - Reassigns the customer service rep based on
assignment rules for the given inputs.
      **
      ** IN parameters
      **    CUSTOMER - sold to org id of the customer
      **    CUSTSERVICEREP - user_id of the customer service rep
      **    OVERRIDE - boolean value for override existing csr
      ** OUT parameters
      **    ERRBUF - error
      **    RETCODE - return status
      **
      **   1) prepare sql statement based on given parameters
      **   2) open cursor with prepared sql statement
      **   3) If override is yes, update the existing csr to null
      **   4) Iterate over the cursor and update the csr
      */
```

```
    PROCEDURE assign_csr (ERRBUF              OUT NOCOPY VARCHAR2,
                          RETCODE             OUT NOCOPY VARCHAR2,
                          CUSTOMER                       NUMBER,
                          CUSTSERVICEREP                 NUMBER,
                          OVERRIDE                       VARCHAR2)
    IS
        l_sold_to_org_id         NUMBER;
        l_customer_service_rep   NUMBER;
        l_min_line_id            NUMBER;
        l_inventory_item_id      NUMBER;
        l_csr_user_id            NUMBER;
        order_header_id          NUMBER;
        l_override               VARCHAR2 (10);
        orders_to_process        SYS_REFCURSOR;
        l_error                  BOOLEAN := FALSE;
        l_headers_all            header_csr_tbl;
        l_ECC_BULK_LOAD_SIZE     NUMBER
            := NVL (fnd_profile.Value_wnps ('OM_ECC_BULK_LOAD_SIZE'),
1000);
        l_query_string           VARCHAR2 (1000)
            := 'select order_number,header_id,sold_to_org_id from
oe_order_headers where NVL(transaction_phase_code, ''F'') = ''F'' and
open_flag =''Y''';
    BEGIN
        l_sold_to_org_id := CUSTOMER;
        l_customer_service_rep := CUSTSERVICEREP;
        l_override := OVERRIDE;

        mo_global.init ('ONT');
        mo_global.set_policy_context ('M', NULL);

        --insert_debug('l_sold_to_org_id :'||l_sold_to_org_id ||'-
l_customer_service_rep :'||l_customer_service_rep|| ' - l_override
:'||l_override);
        fnd_file.put_line (
            FND_FILE.LOG,
               'l_sold_to_org_id :'
            || l_sold_to_org_id
            || '- l_customer_service_rep :'
            || l_customer_service_rep
            || ' - l_override :'
            || l_override);

        IF l_customer_service_rep IS NOT NULL
        THEN
            IF l_override = 'N'
            THEN
                l_query_string := '';
                fnd_file.put_line (
                    FND_FILE.OUTPUT,
                    fnd_message.get_string ('ONT',
'ONT_ECC_OVERRIDE_NO'));
                fnd_file.put_line (
                    FND_FILE.LOG,
                    fnd_message.get_string ('ONT',
'ONT_ECC_OVERRIDE_NO'));
                l_error := TRUE;
                RETCODE := FND_API.G_RET_STS_ERROR;
                l_error := TRUE;
            ELSE
                l_query_string :=
                       l_query_string
                    || ' and csr_user_id = '
                    || l_customer_service_rep;               -- csr
+override
```

```
IF l_sold_to_org_id IS NOT NULL
                THEN
                    l_query_string :=
                           l_query_string
                        || ' and  sold_to_org_id ='
                        || l_sold_to_org_id;             --
csr+override+cust
                END IF;
            END IF;
        ELSIF l_sold_to_org_id IS NOT NULL
        THEN
            l_query_string :=
                   l_query_string
                || '  and  sold_to_org_id ='
                || l_sold_to_org_id
                || ' ';                                        --
cust+override

            IF l_override = 'N'
            THEN
                l_query_string :=
                    l_query_string || ' and csr_user_id is null'; --
only cust
            END IF;
        ELSIF l_override = 'N'
        THEN
            l_query_string := l_query_string || ' and csr_user_id is
null'; -- none
        END IF;

        --insert_debug('l_query_string :'||l_query_string);
        fnd_file.put_line (FND_FILE.LOG,
                           'l_query_string :' || l_query_string);

        IF l_error <> TRUE
        THEN
            OPEN orders_to_process FOR l_query_string;

            LOOP
                FETCH orders_to_process
                    BULK COLLECT INTO l_headers_all
                    LIMIT l_ECC_BULK_LOAD_SIZE;

                EXIT WHEN l_headers_all.COUNT = 0;

                --insert_debug('l_headers_all.COUNT is:'||l_headers_all.
COUNT);
                fnd_file.put_line (
                    FND_FILE.LOG,
                    'l_headers_all.COUNT is:' || l_headers_all.COUNT);

                /* Audit issue - should not update last_update_date for
all the records.
                IF l_override = 'Y' THEN
                    FORALL indx IN 1 .. l_headers_all.COUNT
                        update oe_order_headers_all
                        set csr_user_id = null,
   LAST_UPDATED_BY     = FND_GLOBAL.USER_ID,
                        LAST_UPDATE_DATE    = sysdate,
                        LAST_UPDATE_LOGIN   = FND_GLOBAL.USER_ID
                        where header_id = l_headers_all(indx).header_id;
                END IF;
                */

                FOR i IN l_headers_all.FIRST .. l_headers_all.LAST
                LOOP
```

```
l_inventory_item_id := 0;
                    l_sold_to_org_id := l_headers_all (i).
sold_to_org_id;

                    --insert_debug('order_header_id :'||l_headers_all
(i).header_id);

                    BEGIN
                        SELECT l.inventory_item_id
                          INTO l_inventory_item_id
                          FROM oe_order_lines_all  l,
                               (SELECT MIN (line_id)     min_line_id
                                  FROM oe_order_lines_all
                                 WHERE header_id =
                                       l_headers_all (i).header_id)
l_min
                         WHERE l.line_id = l_min.min_line_id;
                    EXCEPTION
                        WHEN OTHERS
                        THEN
                            l_inventory_item_id := -1;
                            --insert_debug(' sql error for order : '||
l_headers_all(i).order_number);
                            fnd_file.put_line (
                                FND_FILE.LOG,
                                   ' sql error for order : '
                                || l_headers_all (i).order_number);
                    END;

                    --insert_debug('inventory_item_id
:'||l_inventory_item_id ||'l_sold_to_org_id:'||l_sold_to_org_id);
                    fnd_file.put_line (
                        FND_FILE.LOG,
                           'inventory_item_id :'
                        || l_inventory_item_id
                        || ' - l_sold_to_org_id:'
                        || l_sold_to_org_id
                        || ' - order_header_id :'
                        || l_headers_all (i).header_id);

                    IF l_inventory_item_id > 0
                    THEN
                        l_csr_user_id :=
                            oe_ecc_util_pvt.get_csr_details (
                                l_sold_to_org_id,
                                l_inventory_item_id);

                        --insert_debug('updating l_csr_user_id
:'||l_csr_user_id ||' for order number :'||l_headers_all(i).
order_number);
                        -- should not update if l_csr_user_id same as on
the order
                        BEGIN
                            UPDATE oe_order_headers_all
                               SET csr_user_id = l_csr_user_id,
                                   LAST_UPDATED_BY = FND_GLOBAL.USER_ID,
                                   LAST_UPDATE_DATE = SYSDATE,
                                   LAST_UPDATE_LOGIN = FND_GLOBAL.
USER_ID
                             WHERE     header_id =
                                       l_headers_all (i).header_id
                                   AND NVL (csr_user_id, 0) <>
                                       NVL (l_csr_user_id, 0);
                        EXCEPTION
                            WHEN APP_EXCEPTIONS.RECORD_LOCK_EXCEPTION
                            THEN
```

```
--insert_debug('others exception:'||sqlerrm);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                           'RECORD_LOCK_EXCEPTION
exception:'
                                        || SQLERRM);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                           'RECORD_LOCK_EXCEPTION exception:
error occured for '
                                        || l_headers_all (i).header_id
                                        || '--'
                                        || l_headers_all (i).order_number);
                                WHEN OTHERS
                                THEN
                                    --insert_debug('others
exception:'||sqlerrm);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                        'others exception:' || SQLERRM);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                           'others exception: error occured
for '
                                        || l_headers_all (i).header_id
                                        || '--'
                                        || l_headers_all (i).order_number);
                            END;
                        ELSIF l_override = 'Y'
                        THEN
                            -- update csr to null only if all lines are
deleted and csr it is not null
                            BEGIN
                                UPDATE oe_order_headers_all
                                   SET csr_user_id = NULL,
                                       LAST_UPDATED_BY = FND_GLOBAL.USER_ID,
                                       LAST_UPDATE_DATE = SYSDATE,
                                       LAST_UPDATE_LOGIN = FND_GLOBAL.
USER_ID
                                 WHERE     header_id =
                                           l_headers_all (i).header_id
                                       AND csr_user_id IS NOT NULL;
                            EXCEPTION
                                WHEN APP_EXCEPTIONS.RECORD_LOCK_EXCEPTION
                                THEN
                                    --insert_debug('others
exception:'||sqlerrm);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                           'RECORD_LOCK_EXCEPTION
exception:'
                                        || SQLERRM);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                           'RECORD_LOCK_EXCEPTION exception:
error occured for '
                                        || l_headers_all (i).header_id
                                        || '--'
                                        || l_headers_all (i).order_number);
                                WHEN OTHERS
                                THEN
                                    --insert_debug('others
exception:'||sqlerrm);
                                    fnd_file.put_line (
                                        FND_FILE.LOG,
                                        'others exception:' || SQLERRM);
```

```
fnd_file.put_line (
                                    FND_FILE.LOG,
                                       'others exception: error occured
for '
                                    || l_headers_all (i).header_id
                                    || '--'
                                    || l_headers_all (i).order_number);
                        END;
                    END IF;
                END LOOP;

                COMMIT;
            END LOOP;

            CLOSE orders_to_process;

            RETCODE := FND_API.G_RET_STS_SUCCESS;
            -- COMMIT; -- should we commit after every update in inner
loop.
            fnd_file.put_line (FND_FILE.OUTPUT,
                               'Program completed successfully');
            fnd_file.put_line (FND_FILE.LOG,
                               'Program completed successfully');
        END IF;
    EXCEPTION
        WHEN OTHERS
        THEN
            fnd_file.put_line (
                FND_FILE.OUTPUT,
                'OTHERS exception while submitting : ' || SQLERRM);
            fnd_file.put_line (
                FND_FILE.LOG,
                'OTHERS exception while submitting : ' || SQLERRM);
            RETCODE := FND_API.G_RET_STS_ERROR;
    END assign_csr;
/* ECC V3 ER end */

END OE_ECC_UTIL_PVT;
/
```

B

Data Load And Metadata Load Signature

Data Load And Metadata Load Signature

```
CREATE OR REPLACE PACKAGE APPS.icx_ecc_util_pvt AUTHID CURRENT_USER AS
/* $Header: ICXECCUS.pls 120.0.1 2018/11/29 11:10:32 noship $ */
g_icx_ecc_ins_op CONSTANT VARCHAR2(30) := 'INSERT'; g_icx_ecc_upd_op
CONSTANT VARCHAR2(30) := 'UPDATE'; g_icx_ecc_del_op CONSTANT VARCHAR2
(30) := 'DELETE'; g_icx_ecc_ups_op CONSTANT VARCHAR2(30) := 'UPSERT';
g_icx_exc_op CONSTANT VARCHAR2(30) := 'EXECUTE'; g_icx_dsc_op CONSTANT
VARCHAR2(30) := 'DESCRIBE';
g_icx_dsc_att_op CONSTANT VARCHAR2(30) := 'DESCRIBE|ATTRIBUTE';
g_icx_dsc_pre_op CONSTANT VARCHAR2(30) := 'DESCRIBE|PRECEDENCE_RULE';
g_icx_exc_op CONSTANT VARCHAR2(30) := 'EXECUTE'; g_icx_exc_att_op
CONSTANT VARCHAR2(30) := 'EXECUTE|ATTRIBUTE';
g_icx_exc_pre_op CONSTANT VARCHAR2(30) := 'EXECUTE|PRECEDENCE_RULE';
/*
** GET_ECC_DATA_LOAD_INFO - Retrieves the data load details for the
given data set key
**
** IN parameters
** p_dataset_key - data set key
** p_load_type - Indicating Full or incremental load (F/I/Custom).For
custom there are no data load rules defined
** p_ds_last_success_run - Returns the last successful etl run for the
data set key
** OUT parameters
** x_ecc_ds_meta_rec - ecc_ds_meta_rec
** x_return_status - return status
*/
--------------------------------------------------------------------------
PROCEDURE get_ecc_data_load_info ( p_dataset_key IN VARCHAR2,
p_load_type  IN VARCHAR2, p_ds_last_success_run  IN TIMESTAMP,
p_languages IN VARCHAR2,
p_addl_params IN ecc_sec_field_values, x_ecc_ds_meta_rec OUT NOCOPY
ecc_ds_meta_rec, x_return_status OUT NOCOPY VARCHAR2
);
--------------------------------------------------------------------------
/*
** GET_SQL_TEXT - Retrieves the sql text for a data set
** IN parameters
** p_dataset_key - data set key
** p_load_type - Indicating Full or incremental load (F/I/Custom).For
custom there are no data load rules defined
** p_operation - Indicates the operations INSERT,UPDATE,DELETE and
UPSERT
** p_data_type - Indicates the type of data
** p_ds_last_success_run - Returns the last successful etl run for the
data set key
** RETURNS sql text
*/
FUNCTION get_sql_text (
p_dataset_key IN VARCHAR2,
p_load_type IN VARCHAR2,
p_operation IN VARCHAR2,
p_data_type IN VARCHAR2, p_ds_last_success_run IN TIMESTAMP, p_languages
IN VARCHAR2
) RETURN CLOB;
PROCEDURE create_ecc_desc_view ( p_category_id IN NUMBER
);
PROCEDURE create_ecc_desc_views; PROCEDURE drop_ecc_desc_views;
PROCEDURE get_desc_metadata_load_info (
p_dataset_key IN VARCHAR2,
p_dataset_attrs IN ecc_sec_field_values DEFAULT NULL, p_languages  IN
VARCHAR2,
x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec, x_return_status OUT NOCOPY
```

```
VARCHAR2
);

FUNCTION makencname ( p_attribute_name IN VARCHAR2
) RETURN VARCHAR2;
PROCEDURE submit_ecc_data_load (
errbuf OUT NOCOPY VARCHAR2,
retcode OUT NOCOPY VARCHAR2,
p_system_name IN VARCHAR2, p_load_type IN VARCHAR2,
p_languages IN VARCHAR2,
p_log_level IN VARCHAR2, p_trace_enabled IN VARCHAR2
);
END icx_ecc_util_pvt;
/
```

C

Sample Concurrent Program

Sample Concurrent Program

```
PROCEDURE SUBMIT_ECC_DATA_LOAD(
     ERRBUF OUT NOCOPY VARCHAR2 ,
     RETCODE OUT NOCOPY VARCHAR2 ,
     p_system_name IN VARCHAR2,
     p_load_type IN VARCHAR2,
     p_languages IN VARCHAR2,
     p_log_level IN VARCHAR2, p_trace_enabled IN VARCHAR2
 ) IS
  l_app_short_name varchar2(10) := 'icx'; l_request_id number;
  l_req_data varchar2(10); l_req_data1 varchar2(10); l_load_type
varchar2(200);
 BEGIN

  l_req_data := fnd_conc_global.request_data;
  SELECT decode(p_load_type, 'FULL_LOAD', 'METADATA_LOAD', p_load_type)
INTO l_load_type FROM dual;

  IF l_req_data ='END' THEN fnd_file.put_line(FND_FILE.OUTPUT,'Executed
the sub request: ICX
ECC Data Load for load type : ' || p_load_type); fnd_file.put_line
(FND_FILE.LOG,'Executed the sub request: ICX ECC
Data Load for load type : ' || p_load_type); RETURN;
 END IF;

  IF (l_req_data ='END-META' and p_load_type = 'FULL_LOAD') THEN
     fnd_file.put_line(FND_FILE.OUTPUT,'Executed the sub request:
ICX ECC Data Load for load type : ' || l_load_type); fnd_file.put_line
(FND_FILE.LOG,'Executed the sub request: ICX
ECC Data Load for load type : ' || l_load_type); l_request_id :=
      fnd_request.submit_request(
          application => 'FND',
          program => 'ECCRUNDL',
          description => 'ECC - Run Data Load',
          start_time => sysdate,
          argument1 => p_system_name,
          argument2 => l_app_short_name,
          argument3 => null,
          argument4 => p_load_type,
          argument5 => p_languages,
          argument6 => p_trace_enabled,
          argument7 => p_log_level,
          sub_request => true);
      fnd_file.put_line(FND_FILE.OUTPUT,'ECC Load for load type : '
|| p_load_type || ' Job Request ID:'||l_request_id); fnd_file.put_line
(FND_FILE.LOG,'ECC Load for load type : ' ||
p_load_type || ' Job Request ID:'||l_request_id);

      IF l_request_id = 0 THEN
         --
         -- If request submission failed, exit with error.
         --


  ELSE
--

 -        - Here we set the globals to put the program into the
       -- PAUSED status on exit, and to save the state in
  -- request_data.
         --
         fnd_conc_global.set_req_globals(conc_status => 'PAUSED',
request_data => 'END');
```

```
        ERRBUF := 'Sub-Request submitted!'; RETCODE := 0 ;

        END IF;
        RETURN;
  END IF;
   -- Submit l_load_type which can be either METADATA_LOAD or
INCREMENTAL_LOAD.
   l_request_id := fnd_request.submit_request(
            application => 'FND', program => 'ECCRUNDL',
            description => 'ECC - Run Data Load',
            start_time => sysdate,
            argument1 => p_system_name,
            argument2 => l_app_short_name,
            argument3 => null,
            argument4 => l_load_type,
            argument5 => p_languages,
            argument6 => p_trace_enabled,
            argument7 => p_log_level,
            sub_request => true);

   fnd_file.put_line(FND_FILE.OUTPUT,'ECC Load for load type : ' ||
l_load_type || ' Job Request ID:'||l_request_id);
   fnd_file.put_line(FND_FILE.LOG,'ECC Load for load type : ' ||
l_load_type || ' Job Request ID:'||l_request_id);

     IF l_request_id = 0 THEN
      --
      -- If request submission failed, exit with error.
      --
      ERRBUF := fnd_message.get;
      RETCODE := 2;
     ELSE
      --
      -- Here we set the globals to put the program into the
      -- PAUSED status on exit, and to save the state in
      -- request_data.
      --


       IF(l_load_type = 'METADATA_LOAD' AND p_load_type='FULL_LOAD')
   THEN
          l_req_data1 := 'END-META'; ELSE
          l_req_data1 := 'END'; END IF;

      fnd_conc_global.set_req_globals(conc_status => 'PAUSED',
request_data => l_req_data1);

      ERRBUF := 'Sub-Request submitted!'; RETCODE := 0 ;

     END IF;
    RETURN;
   EXCEPTION
   WHEN OTHERS
    THEN RETCODE
    := 1;
    fnd_file.put_line(FND_FILE.OUTPUT,'OTHERS exception while submitting
    : ICX ECC Data Load' || sqlerrm); fnd_file.put_line(FND_FILE.
LOG,'OTHERS exception while submitting :
    ICX ECC Data Load' || sqlerrm);
      ERRBUF := fnd_message.get; RETCODE := 2
END SUBMIT_ECC_DATA_LOAD;
```

D

Metadata and Data Load Procedure

Code Example for Designing a Metadata Load Procedure

```
create or replace PACKAGE      CUSTOM_AR_ECC_UTIL_PVT AUTHID
CURRENT_USER AS
G_AR_ECC_INS_OP CONSTANT VARCHAR2(30) := 'INSERT';
G_AR_ECC_UPD_OP CONSTANT VARCHAR2(30) := 'UPDATE';
G_AR_ECC_DEL_OP CONSTANT VARCHAR2(30) := 'DELETE';
G_AR_ECC_UPS_OP CONSTANT VARCHAR2(30) := 'UPSERT';
G_AR_ECC_QRY_OP CONSTANT VARCHAR2(30) := 'DELETE_BY_QUERY';
CUSTOM_ECC_ATTRIBUTES     CONSTANT VARCHAR2(30) := 'EXECUTE|ATTRIBUTE';
PROCEDURE CUSTOM_GET_ECC_DATA_LOAD_INFO(p_dataset_key IN VARCHAR2,
             p_load_type IN VARCHAR2,
    p_ds_last_success_run IN DATE,
    p_languages IN VARCHAR2,
             p_addl_params IN ecc_sec_field_values,
             x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec,
       x_return_status OUT NOCOPY VARCHAR2);
------------------------------------------------------------------------
-----------------------------------
PROCEDURE GET_DESC_METADATA_LOAD_INFO (
        p_dataset_key       IN  VARCHAR2,
        p_dataset_attrs     IN  ecc_sec_field_values DEFAULT NULL,
        p_languages         IN  VARCHAR2,
        x_ecc_ds_meta_rec   OUT NOCOPY ecc_ds_meta_rec,
        x_return_status     OUT NOCOPY VARCHAR2
  );
END CUSTOM_AR_ECC_UTIL_PVT;

create or replace PACKAGE BODY      CUSTOM_AR_ECC_UTIL_PVT AS
PROCEDURE CUSTOM_GET_ECC_DATA_LOAD_INFO(p_dataset_key IN VARCHAR2,
             p_load_type IN VARCHAR2,
    p_ds_last_success_run IN DATE,
    p_languages IN VARCHAR2,
    p_addl_params IN ecc_sec_field_values,
             x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec,
       x_return_status OUT NOCOPY VARCHAR2)
IS
v_for_lang_pivot_clause varchar2(400) := FND_ECC_UTIL_MLS_PVT.
GEN_ECC_MLS_PIVOT_FOR_LANG_CL(p_languages);
v_lang_in varchar2(400) := FND_ECC_UTIL_MLS_PVT.GEN_ECC_MLS_WHERE_CL
(p_languages);
l_log_module       CONSTANT VARCHAR2(255) := 'ar.plsql.
AR_ECC_UTIL_PVT_CUSTOM.GET_ECC_DATA_LOAD_INFO';
  l_ar_installments  VARCHAR2(30000) := 'SELECT * from (SELECT /*+
leading ( trx_v.temp.trx_t trx_v.temp.ps trx_v.temp.trx) full (trx_v.
temp.trx_t ) cardinality (trx_v.temp.trx_t 10) use_nl(trx_v.temp.trx) */
  trx_v.ECC_SPEC_ID, trx_v.COMMENTS CUSTOM_ATTRIBUTE
  FROM ari_ecc_trx_v            trx_v
  where trx_v.language in ('||v_lang_in||')) ';
  li_ar_installments  VARCHAR2(30000) := 'SELECT * from (SELECT /*+
leading ( trx_v.temp.trx_t trx_v.temp.ps trx_v.temp.trx) full (trx_v.
temp.trx_t ) cardinality (trx_v.temp.trx_t 10) use_nl(trx_v.temp.trx) */
  trx_v.ECC_SPEC_ID, trx_v.COMMENTS CUSTOM_ATTRIBUTE
  FROM ari_ecc_trx_v            trx_v
  where trx_v.language in ('||v_lang_in||')) ';
query_det_arr  ecc_query_det_arr_type := ecc_query_det_arr_type(null);
l_return_status         VARCHAR2(1);
BEGIN
IF (p_load_type = 'FULL_LOAD' ) THEN
 IF (p_dataset_key='ar-copy') THEN
  query_det_arr.extend(1);
     query_det_arr(1) := ecc_query_det_rec(l_ar_installments,
G_AR_ECC_INS_OP);
          x_ecc_ds_meta_rec  :=ecc_ds_meta_rec(p_dataset_key,
```

```
query_det_arr);
          x_return_status  :='S';
    END IF;
ELSIF(p_load_type ='INCREMENTAL_LOAD') THEN
 IF (p_dataset_key='ar-copy') THEN
  query_det_arr.extend(1);
  query_det_arr(1) := ecc_query_det_rec(li_ar_installments,
G_AR_ECC_INS_OP);
          x_ecc_ds_meta_rec  :=ecc_ds_meta_rec(p_dataset_key,
query_det_arr);
          x_return_status  :='S';
   END IF;
 END IF;
END CUSTOM_GET_ECC_DATA_LOAD_INFO;
 PROCEDURE GET_DESC_METADATA_LOAD_INFO (
        p_dataset_key       IN  VARCHAR2,
        p_dataset_attrs     IN  ecc_sec_field_values DEFAULT NULL,
        p_languages         IN  VARCHAR2,
        x_ecc_ds_meta_rec   OUT NOCOPY ecc_ds_meta_rec,
        x_return_status     OUT NOCOPY VARCHAR2
  ) IS
  query_det_arr           ecc_query_det_arr_type :=
ecc_query_det_arr_type(NULL);
  l_dff_query_tbl         FND_ECC_DFF_UTIL.dff_query_tbl;
  vrow                    FND_ECC_DFF_UTIL.dff_query_rec;
  l_return_status         VARCHAR2(1) := 'S';
  l_dff1_cnt              NUMBER := 0;
  l_dff2_cnt              NUMBER := 0;
  l_apps_schema_name      VARCHAR2(10);
  l_log_module       CONSTANT VARCHAR2(255) := 'ar.plsql.
AR_ECC_UTIL_PVT_CUSTOM.GET_DESC_METADATA_LOAD_INFO';
  l_ar_metadata  VARCHAR2(30000) := ' SELECT '||'''XYZ_'''||'||
COLUMN_NAME FROM ALL_TAB_COLUMNS WHERE TABLE_NAME
=''RA_CUSTOMER_TRX_ALL_DFV'' ';
   BEGIN
 if p_dataset_key = 'ar-copy' then
   query_det_arr(1) := ecc_query_det_rec(l_ar_metadata,
CUSTOM_ECC_ATTRIBUTES);
     x_ecc_ds_meta_rec  :=ecc_ds_meta_rec(p_dataset_key,query_det_arr);
     x_return_status  :='S';
 end if;
 END GET_DESC_METADATA_LOAD_INFO;
END CUSTOM_AR_ECC_UTIL_PVT;
```

Code Example for Modifying a Data Load Procedure

```
create or replace PACKAGE      CUSTOM_AR_ECC_UTIL_PVT AUTHID
CURRENT_USER AS
G_AR_ECC_INS_OP CONSTANT VARCHAR2(30) := 'INSERT';
G_AR_ECC_UPD_OP CONSTANT VARCHAR2(30) := 'UPDATE';
G_AR_ECC_DEL_OP CONSTANT VARCHAR2(30) := 'DELETE';
G_AR_ECC_UPS_OP CONSTANT VARCHAR2(30) := 'UPSERT';
G_AR_ECC_QRY_OP CONSTANT VARCHAR2(30) := 'DELETE_BY_QUERY';PROCEDURE
CUSTOM_GET_ECC_DATA_LOAD_INFO(p_dataset_key IN VARCHAR2,
             p_load_type IN VARCHAR2,
 p_ds_last_success_run IN DATE,
 p_languages IN VARCHAR2,
             p_addl_params IN ecc_sec_field_values,
             x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec,
     x_return_status OUT NOCOPY VARCHAR2);
------------------------------------------------------------------------
------------------------------------ CUSTOM_AR_ECC_UTIL_PVT;

create or replace PACKAGE BODY      CUSTOM_AR_ECC_UTIL_PVT AS
PROCEDURE CUSTOM_GET_ECC_DATA_LOAD_INFO(p_dataset_key IN VARCHAR2,
             p_load_type IN VARCHAR2,
    p_ds_last_success_run IN DATE,
    p_languages IN VARCHAR2,
    p_addl_params IN ecc_sec_field_values,
             x_ecc_ds_meta_rec OUT NOCOPY ecc_ds_meta_rec,
       x_return_status OUT NOCOPY VARCHAR2)
IS
v_for_lang_pivot_clause varchar2(400) := FND_ECC_UTIL_MLS_PVT.
GEN_ECC_MLS_PIVOT_FOR_LANG_CL(p_languages);
v_lang_in varchar2(400) := FND_ECC_UTIL_MLS_PVT.GEN_ECC_MLS_WHERE_CL
(p_languages);
l_log_module       CONSTANT VARCHAR2(255) := 'ar.plsql.
AR_ECC_UTIL_PVT_CUSTOM.GET_ECC_DATA_LOAD_INFO';
  l_ar_installments  VARCHAR2(30000) := 'SELECT * from (SELECT /*+
leading ( trx_v.temp.trx_t trx_v.temp.ps trx_v.temp.trx) full (trx_v.
temp.trx_t ) cardinality (trx_v.temp.trx_t 10) use_nl(trx_v.temp.trx) */
  trx_v.ECC_SPEC_ID, trx_v.COMMENTS CUSTOM_ATTRIBUTE
  FROM ari_ecc_trx_v            trx_v
  where trx_v.language in ('||v_lang_in||')) ';
  li_ar_installments  VARCHAR2(30000) := 'SELECT * from (SELECT /*+
leading ( trx_v.temp.trx_t trx_v.temp.ps trx_v.temp.trx) full (trx_v.
temp.trx_t ) cardinality (trx_v.temp.trx_t 10) use_nl(trx_v.temp.trx) */
  trx_v.ECC_SPEC_ID, trx_v.COMMENTS CUSTOM_ATTRIBUTE
  FROM ari_ecc_trx_v            trx_v
  where trx_v.language in ('||v_lang_in||')) ';
query_det_arr  ecc_query_det_arr_type := ecc_query_det_arr_type(null);
l_return_status         VARCHAR2(1);
BEGIN
IF (p_load_type = 'FULL_LOAD' ) THEN

 IF (p_dataset_key='ar-copy') THEN
  query_det_arr.extend(1);
     query_det_arr(1) := ecc_query_det_rec(l_ar_installments,
G_AR_ECC_INS_OP);
          x_ecc_ds_meta_rec  :=ecc_ds_meta_rec(p_dataset_key,
query_det_arr);
          x_return_status  :='S';
    END IF;

ELSIF(p_load_type ='INCREMENTAL_LOAD') THEN
 IF (p_dataset_key='ar-copy') THEN
  query_det_arr.extend(1);
  query_det_arr(6) := ecc_query_det_rec(li_ar_installments,
```

```
G_AR_ECC_INS_OP);
          x_ecc_ds_meta_rec  :=ecc_ds_meta_rec(p_dataset_key,
query_det_arr);
          x_return_status  :='S';
   END IF;
 END IF;
END CUSTOM_GET_ECC_DATA_LOAD_INFO;
END CUSTOM_AR_ECC_UTIL_PVT;
```

E

Sample Custom Security Handler

Sample Custom Security Handler

```
create or replace PACKAGE CUSTOM_AP_ECC_DS_PKG_PUB  as

procedure GetFilterAttributeValues (
                 x_dataset_key     IN VARCHAR2             ,
                 x_user_id         in number               ,
                  x_org_id          in number               ,
                 x_resp_id         in number               ,
                 x_resp_app_id     in number               ,
                 x_sec_group_id    in number               ,
                 x_params          in ecc_sec_field_values DEFAULT NULL,
                   x_return_status  out nocopy varchar2             ,
                   x_return_message out nocopy varchar2             ,
                   x_sec_filter     out nocopy CLOB    );

 PROCEDURE LogStatement(p_pkgname IN VARCHAR2,
                     p_message IN VARCHAR2);

END CUSTOM_AP_ECC_DS_PKG_PUB

Package Body:

create or replace PACKAGE BODY CUSTOM_AP_ECC_DS_PKG_PUB  as

procedure GetFilterAttributeValues (
                 x_dataset_key     IN VARCHAR2             ,
                 x_user_id         in number               ,
                  x_org_id          in number               ,
                 x_resp_id         in number               ,
                 x_resp_app_id     in number               ,
                 x_sec_group_id    in number               ,
                 x_params          in ecc_sec_field_values DEFAULT NULL,
                   x_return_status  out nocopy varchar2             ,
                   x_return_message out nocopy varchar2             ,
                   x_sec_filter     out nocopy CLOB    ) IS


     x_return_string CLOB;
     l_sec_list_values ecc_sec_field_values;
     l_ou_tab         mo_global.OrgIdTab;
     l_tab            icx_tbl_varchar240;
     l_count          NUMBER := 0;
     l_application_id NUMBER := NULL;
     x_sec_list_values   ecc_sec_field_values;
       l_sec_filter      CLOB;
  debug_info                VARCHAR2(100);
       l_profile_id       varchar2(1000);
 BEGIN

     /* This method would be called by the ECC Security manager
      * it would pass whatever parameter values are available in the
context
      * product teams need to write individual methods to handle the
specific
      * data security scenarios for every dataset
      * dataset_key would be the first parameter passed
      * The next 5 parameters provide the EBS context to construct the
security filters
      * The x_params is an array of VARCHAR2(100)
      * x_return_status should return
      *     'UNEXPECTED ERROR' in case an exception is encountered ,
      *     'NO_SECURITY'      if no security should be applied to the
dataset
```

```
*     'EXPECTED ERROR'   in case of an expected error ,
      *                        such as an incomplete setup or any
condition where security filter cannot be constructed
      * x_return_message should return the error message in case the
x_return_status is 'UNEXPECTED ERROR' or 'EXPECTED ERROR'
      * the x_sec_filter would have the final filter as a json string
which be a return value of the call to the methods in
ecc_datasecurity_pkg
      */

       LogStatement('CUSTOM_AP_ECC_DS_PKG_PUB.GetFilterAttributeValues',
'In the Payables datasecurity procedure');
    LogStatement('CUSTOM_AP_ECC_DS_PKG_PUB.GetFilterAttributeValues',
'dataset = ' || x_dataset_key  ||' respid = ' || x_resp_id ||  'userid=
' || x_user_id||' respid app ud= ' || x_resp_app_id );


         l_sec_filter := NULL;

         l_sec_list_values  := NEW ecc_sec_field_values(null);

    Fnd_Global.Apps_Initialize(
    x_user_id,
    x_resp_id,
    x_resp_app_id,
         x_sec_group_id
    );

     MO_GLOBAL.init('SQLAP');
     mo_global.set_policy_context('M', -1);

             // customer need to prepare below query as per their
requirement to get filter values.
         select user_id bulk_collect into l_sec_list_values from
per_organization_list;

       l_sec_filter := FND_ECC_DATASECURITY_PKG_PUB.
CREATERECORDFILTEREXPROR('USER_ID',l_sec_list_values,x_return_status ,
x_return_message);
       x_sec_filter := x_sec_filter||l_sec_filter;

   End If;

   LogStatement('CUSTOM_AP_ECC_DS_PKG_PUB.
GetFilterAttributeValues','List of values:' || x_sec_filter);

  x_return_status := FND_ECC_DATASECURITY_PKG_PUB.ECC_SEC_SUCCESS;
  x_return_message := 'Successful completion';

 EXCEPTION
     WHEN OTHERS THEN
       LogStatement('GetFilterAttributeValues','in the Payables datasec
procedure - exception' || SQLERRM);
       x_sec_filter := null;
       x_return_status := FND_ECC_DATASECURITY_PKG_PUB.
ECC_SEC_UNEXP_ERR;
       x_return_message := substr(sqlerrm,1,4000);
       RAISE_APPLICATION_ERROR(-20144,'Exception in
CUSTOM_AP_ECC_DS_PKG_PUB ' || Substr(sqlerrm,1,1000));
 END GetFilterAttributeValues;

 PROCEDURE LogStatement(p_pkgname IN VARCHAR2,
                        p_message IN VARCHAR2) IS
 BEGIN
 if (FND_LOG.LEVEL_STATEMENT >= FND_LOG.G_CURRENT_RUNTIME_LEVEL) then
  FND_LOG.STRING(FND_LOG.LEVEL_STATEMENT, p_pkgname, p_message);
```

```
end if;
 END LogStatement;

 END CUSTOM_AP_ECC_DS_PKG_PUB;
```

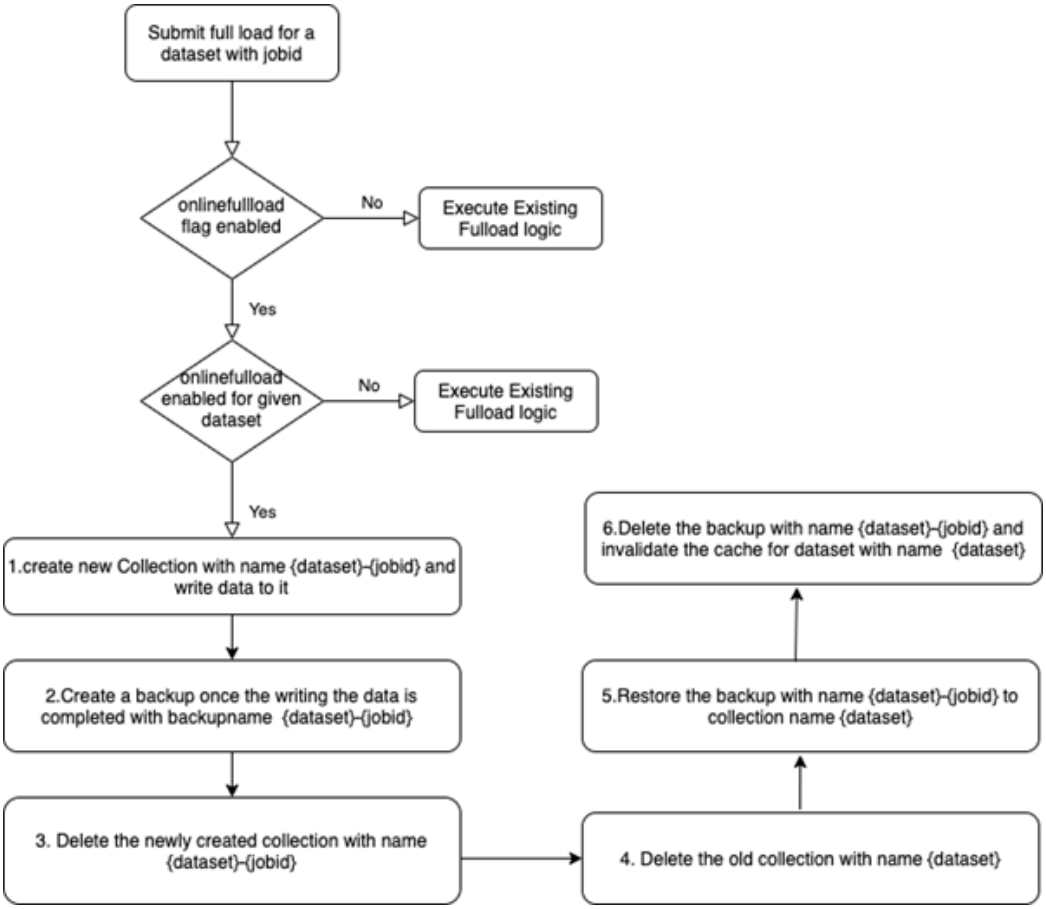
F

Additional Full Load Features

Online Full Load

Oracle Enterprise Command Center Framework supports online full load to allow users to access the dashboard functionality even during full data load execution. This feature can be utilized to greatly reduce the downtime of ECC dashboards because of full data loads. Online full load works as described in the flowchart below:

Online Full Load Flowchart



1. A user submits a full load for a data set with a job ID.
2. If the `onlinefullload` flag is disabled, then the existing full load logic is executed.
3. If the `onlinefullload` flag is not enabled for the given data set, then the existing full load logic is executed.
4. If the `onlinefullload` flag is enabled for the steps above, then the logic for online full load logic is executed.

The steps for online full load are:

1. Create a new collection with the name `<data set>-<job ID>` and write data to it.
2. Create a backup after the writing of the data is completed, with backup name `<data set>-<job ID>`.

3. Delete the newly-created collection with the name `<data set>-<job ID>`
4. Deleted the old collection with the name `<data set>`.
5. Restore the backup with the name `<dataset>-<job ID>` to collection name `<dataset>`.
6. Deleted the backup with the name `<dataset>-<job ID>` and invalidate the cache for the data set with name `<data set>`.

Any errors until step 4 will rollback the changes to the data set and users will still be able to use the current version of the data set. Errors after step 4 will not rollback the changes as the data set is deleted in step 4.

Enabling Online Full Load

Follow the steps below to enable online full load:

1. Set the below properties in configuration properties:
   - Set the property `ecc.onlinefulload.isEnabled` to `true` for enabling the online full load. By default, the property is set to `false`.
   - Add the data sets requiring online full load to property `ecc.onlinefulload.enabled.datasets = <dataset1>,<dataset2>`. Each data set key has to be separated by ','. There are no data sets mentioned by default.
   - Add the location for creating temporary backup in property `ecc.onlinefulload. staging.location = <location of temporary backup for onlinefulload>`. No location is mentioned by default.
2. Restart the ECC managed server to apply the changes.
3. Validate changes by submitting a full load for one of the data sets enabled for online data load.

Sequential Full Load

Sequential load employs a staggered data load mechanism by segregating the data load execution based on unique values of the attribute set as a sequential load controller. The Sequential Full Load feature serializes the data load by languages, generating fine-grained queries that yield better performance compared to the conventional data load process that uses coarse-grained queries.

This feature an improve the data load performance as it provides better flexibility for customers to load only a subset of essential data rather than the full data.

How Sequential Full Load Works

A conventional full load resets all the data to reflect the state of data in Oracle E-Business Suite, whereas an incremental load identifies the last run date at the data set level, and then fetches the incremental changes from that date to update the Enterprise Command Center data set.

Sequential full load is a composite of full data loads for all the identifiers that form the data set and it sequences the full loads per identifier. Sequential full load ensures that the last run date is set at an identifier level so that you can run an incremental load considering this last run date.

Sequential data load also allows you to set a composite identifier basis, in which the full data loads are sequenced.

Sequential data load is triggered from a wrapper concurrent program. The data load request sends the identifier along with the other standard parameters to the central data load concurrent program. The following steps are performed:

1. The last run date for each identifier is gathered. Empty identifiers are treated as a null value.

2. If the last run has no identifier, then the last run date is set for all the identifiers that make a composite identifier.

3. If a data set has been reset after a run for the identifiers, then the last run date would be considered to have a null value.

Enabling Sequential Full Load

The sequential full load feature is controlled by a profile option in Oracle Application Object Library. This feature has to be enabled per each data set's wrapper concurrent program definition using the steps below:

- Add two parameters: Identifier and Set/Reset Data (force flag)

- The Set/Reset Data parameter uses a List of Values with two values: 'Extend Existing Data set' and 'Data set Reset'

This component would be a central component similar to how the load Type Value set is defined. The upper limit on identifier length for each job is 240 characters including the separator character in the case of a composite identifier.

The profile option FND: ECC Sequential Data Load Per Data Set Language (code ECC_DATASETS_SERIAL_LANGS) has the following valid values:

- ALL

- NONE

- Comma-separated list of data set keys

For more information on using profile options, see: User Profiles and Profile Options in Oracle Application Object Library, *Oracle E-Business Suite Setup Guide*.

The wrapper concurrent program has the following characteristics:

- The fields identifier and the parameter `p_force_flag` (for data set reset) for the wrapper concurrent program executable.
  - Example: `p_lastrun_identifier:=p_org_code;`
  - The last run date identifier should be assigned the parameter `p_org_code` itself
- For a composite identifier, the last run identifier should use "|" as separator.

  Note that this separator is applicable only for the Central ECC Data Load job.

For example:

```
PROCEDURE SUBMIT_ECC_DATA_LOAD (
             ERRBUF          OUT NOCOPY VARCHAR2 ,
             RETCODE         OUT NOCOPY VARCHAR2 ,
             p_system_name   IN VARCHAR2,
             p_dataset_key   IN VARCHAR2,
             p_load_type     IN VARCHAR2,
             p_languages     IN VARCHAR2,
             p_log_level     IN VARCHAR2,
             p_trace_enabled IN VARCHAR2,
             p_org_code  IN VARCHAR2,
             p_force_flag IN VARCHAR2,
```

In running the wrapper concurrent program, note the following conditions of the parameters:

- Pass the following parameters in the call to ECC Run Data Load:
  - Identifier as custom parameter
  - Identifier as `p_lastrun_identifier`
  - Parameter `f_force_flag` as the reset flag
- The upper limit on the length of `p_lastrun_identifier` for each job is 240 characters.
- These parameters are optional and useful only for sequential data load.

```
-- Full Load
   fnd_file.put_line(FND_FILE.LOG,'Submitting request: PIM ECC Data Load
for load type l_load_type: 1 || p_load_type)
     IF (p_dataset_key IS NOT NULL ) THEN
        l_request_id := fnd_request.submit_request(
               'FND',
               'ECCRUNDL',
               'ECC - Run Data Load',
               sysdate,
               true,
               p_system_name,
               lapp_short_name,
               NVL(p_dataset_key,null),
               p_load_type,
               p.languages,
               p_trace_enabled,
               p_log_level,
               p_org_code,
               null,
               null,
               null,
               null,
               null,
               null,
               null,
               null,
               null,
               null,
               ull,
               null,
               null,
               null,
               p_lastrun_identifier,
               p_force_flag
               ):
ELSE
```

The ECC Data Load Package includes

- Procedure that gets the identifier passed in the array `p_addl_params`
- Procedure to update the full load and incremental load queries to update the SQL clause

```
PROCEDURE GET_ECC_DATA_LOAD_INFO(p_dataset_key IN VARCHAR2,
                                 p_load_type IN VARCHAR2,
                                 p_ds_last_success_run IN DATE,
                                 p_languages IN VARCHAR2,
                                 p_addl_params IN
ecc_sec_field_values,
                                 x_ecc_ds_meta_rec OUT NOCOPY
ecc_ds_meta_rec,
                                 x_return status OUT NOCOPY
VARCHAR2) IS
...
IF (p_addl_params.exists(1)) THEN
    l_org_code := p_addl_params(l);


...
ELSIF (p_dataset_key = inv-related-item') THEN
l_sql_text:='SELECT *
  FROM ( SELECT
           ECC_SPEC_ID,
           INVENTORY_ITEM_ID,
           ORGANIZATION_ID,
           relationship_type_id,
           ORG_CODE,
           ITEM,
           DESCRIPTION,
           RELATIONSHIP_TYPES,
           RELATED_ITEM,
           LANGUAGE
           DFF_SELECT_TOKEN
       FROM
           MTL_ITEM_ECC_REL_ITEMS_V mtl_v DFF_FROM_TOKEN where
LANGUAGE in ('||v_lang_in [...]
           '|| l_org_code_clause||' DFF_WHERE_TOKEN)
PIVOT (MAX(description) AS DESCRIPTION,MAX(relationship_types) as
RELATIONSHIP_TYPES
FOR language IN('|| v_for_lang_pivot_clause ||'))';
```

- Add/delete query as part of the full load for an identifier. This is required when the data set is not reset but you want to reload the entire data set for an identifier using Full Load.

```
query_det_arr(1) := ecc_query_det_rec(ld_ap_inv_acc,
'DELETE_BY_QUERY');
x_ecc_ds_meta_rec :=ecc_ds_meta_rec(p_dataset_key,query_det_arr);
```

```
ld_ap_inv_acc varchar2(10000) := 'SELECT DISTINCT
    ''ECC_SPEC_ID'' as ATTRIBUTE_NAME,
    aia.invoice_id as ATTRIBUTE_VALUE,
    ''LIKE'' AS OPERATOR
FROM
   ap_invoices_all          aia,
   ap_suppliers             ap,
   ap_supplier_sites_all    pos,
   xla_events               xe,
   XLA_TRANSACTION_ENTITIES XTE,
   xla_ae_headers           xah,
   xla_ae_lines             xal
Where
   ap.vendor_id = aia.vendor_id
       AND aia.vendor_site_id = pos.vendor_site_id
   AND xe.application_id = 200
   AND xe.process_status_code <> ''U''
   AND XTE.APPLICATION_ID = 200
   AND NVL(XTE.SOURCE ID INT 1.-991= AIA.INVOICE ID

...
```

- If the last run date is null for an incremental load, then the query for a full load should be run. If the last run date returned is not null, then the incremental load query should be run.

```
IF (p_load_type = 'FULL_LOAD')
OR ((p_load_type = 'INCREMENTAL_LOAD') AND p_ds_last_success_run is
null) THEN

...
```

G

Enabling Value Set Descriptions

Descriptive Flexfield Value Set Descriptions

The Descriptive Flexfield Value Set Description feature enhances the readability of descriptive flexfield filters in available refinements. Business users get an understandable text instead of a code allowing them to apply the right refinements. The following validation types are supported:

- None
- Independent
- Dependent
- Table

To implement this feature, you first define your independent or dependent values sets or table-validated value sets. Next, update the descriptive flexfield implementation in Oracle Enterprise Command Center Framework (ECC).

Oracle Enterprise Command Center Framework then uses the descriptions of the descriptive flexfield values in the display in the dashboard.

The ECC dashboard behavior is similar to applying a refinement using a value code.

For more information on descriptive flexfields, see: Descriptive Flexfield Concepts, *Oracle E-Business Suite Flexfields Guide*. For more information on value sets, see: Values and Value Sets, *Oracle E-Business Suite Flexfields Guide*.

Enabling Value Set Descriptions

There are no special uptake steps for enabling the feature. The descriptive flexfield implementation enables this feature.

A profile option FND: ECC Disable DFF Value Set Description Load (internal name ECC_DFF_DISABLE_VALUESET_DESC_LOAD) can be used to disable the display of

descriptive value set descriptions. This profile option is owned by Oracle Application Object Library and uses the Security hierarchy type. It is visible and updatable on the site level. The default value is null. You can set its value to a comma-separated list of descriptive flexfield names. By default, there is no descriptive flexfield name set in the profile option value, so the feature is enabled for all descriptive flexfields.

> **Note:** Beginning with V8, Oracle Enterprise Command Center Framework has improved date and number format validation for descriptive flexfield attribute values. The respective records with data that do not have supported formats will cause failures during the data load process, and the data load will be only a partial success.

H

Integration with Oracle Application Express (APEX)

Introduction

Oracle Application Express (APEX) is a rapid web application development tool for Oracle Database. Using only a web browser and limited programming experience, users can develop professional applications that are both fast and secure with built-in features such as user interface themes, navigational controls, form handlers, and flexible reports. By doing so, Oracle Application Express accelerates the application development process.

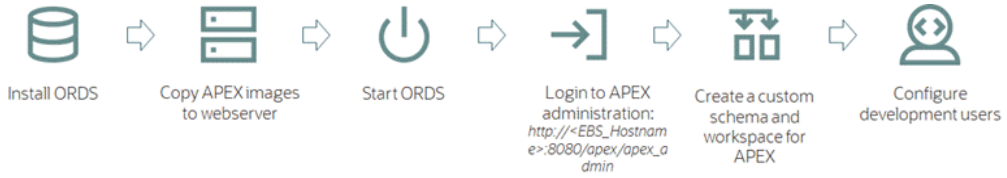
Oracle Enterprise Command Center Framework (ECC) provides operation intelligence dashboards that are action-driven, and customers can take advantage of APEX to design action-focused forms. So, instead of navigating to regular Oracle E-Business Suite forms to perform a required action, users can navigate to the APEX form and perform the action.

Implementation

Refer to *Installing Oracle Enterprise Command Center Framework, Release 12.2*, My Oracle Support Knowledge Document 2495053.1 for instructions on the implementation of Oracle Application Express with Oracle Enterprise Command Center Framework. Also refer to *Extending Oracle E-Business Suite Release 12.1.3 and Above Using Oracle Application Express (APEX)*, My Oracle Support Knowledge Document 1306563.1, and *Oracle Application Express Tutorial: Building an Application* for details on working in APEX.

The implementation process is described at a high level in the following illustration:

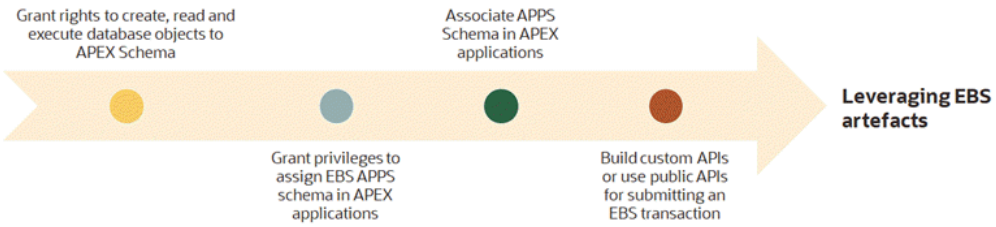
Implementation of Oracle APEX



1. Install Oracle REST Data Services (ORDS).
2. Copy APEX images to the webserver.
3. Start ORDS.
4. Log in to APEX administration:
   `http://<EBS_Hostname>:8080/apex/apex_admin`
5. Create a custom schema and workspace for APEX.
6. Configure development users.

To integrate with Oracle E-Business Suite, the tasks in the following illustration should also be completed:

Integrating Oracle APEX with Oracle E-Business Suite



1. Grant rights to developers to create, read, and execute database objects to APEX schema.
2. Grant privileges to assign the Oracle E-Business Suite APPS schema in APEX applications.
3. Associate the APPS schema in APEX applications.
4. Build custom APIs or use public APIs for submitting an Oracle E-Business Suite transaction.

After performing the above tasks, you can leverage Oracle E-Business Suite artifacts.

Authentication and Authorization

Create an authorization scheme by navigating from Application to Shared Components and then select **Authorization Schemes**.

1. Enter the name of the authorization scheme, `<Authorization Scheme Name>`.
2. Select the scheme type **PL/SQL Function Returning Boolean**.
3. Provide the PL/SQL below inside the PL/SQL Function body:

```
begin
return fnd_ecc_apex_authorization.authorize_apex;
end;
```

4. Provide an error message in case of authentication or authorization failure after the execution of the above PL/SQL.
5. Save the authorization scheme changes.

This authorization scheme validates both the authentication and authorization of the user, so there is no need for explicit configuration for authentication. This API validates against the responsibility of the user accessing the application and then provides the required access.

In the respective pages, select the created scheme as the Authorization Scheme. For Authentication, select **Page is Public** (because authentication is also being validated as part of the authorization scheme).

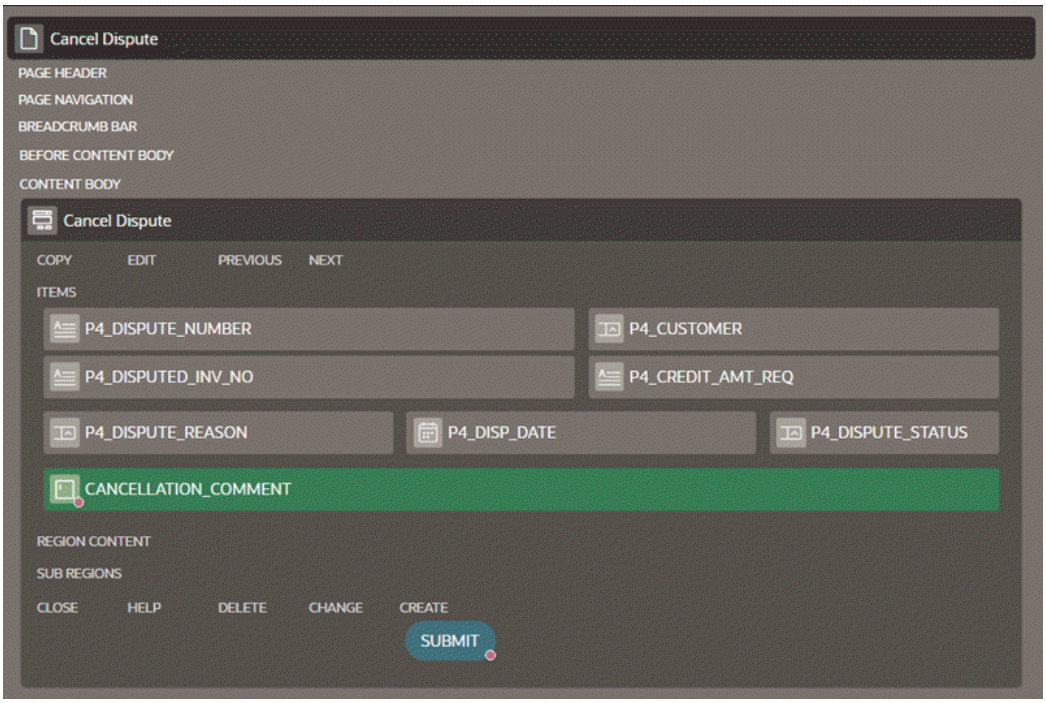
The API used in the authorization scheme retrieves information such as "EBS User ID" and "EBS Language" in addition to performing authentication and authorization.

Building APEX Content

After completing the installation and authorization/authentication, you can start building the application that will contain the pages and forms needed for an action from an Enterprise Command Center dashboard. Use the following steps:

1. Create Fields

Add the required fields and buttons.



For example, there are often disputes in customer invoices due to reasons like incorrect calculations, duplicate line entries, incorrect rates, product damages, and so on. The ECC iReceivables dashboard provides a powerful solution to track customer disputes and cancel the disputes created by the customers.

In this example, we design an APEX form that can be displayed inside an ECC Pop-up window.

The requirements for such an APEX form are:

1. We need to display the details that are required to perform the action; that is, cancel the dispute.

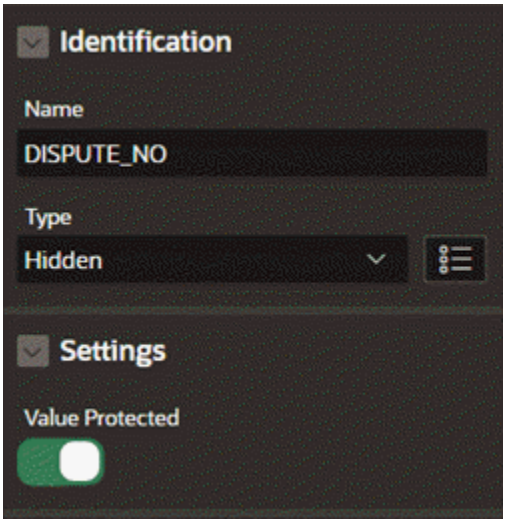
2. The form will include display read-only details such as: Dispute Number, Customer Details, Disputed Invoice Number, Credit Amount Requested, Dispute Reason, Dispute Status, and Dispute Date.

3. A Cancellation Comment field is required where the user enters the comments required to cancel the dispute.

4. A Submit button is required to cancel the dispute.

In addition, the URL Parameter(s) that are used in the ECC hyperlink configuration should be added as the hidden field(s) as these values need not be visible on the APEX page. The following illustration shows an example of a hidden field in a Cancel Dispute form.

Example of a Hidden Field



2. Fetch Data

The data from Oracle E-Business Suite need to be populated in the respective fields, such as in an Oracle Application Framework page. In the case of using APEX, we fetch the details that are required to perform an action as follows:

1. Select the type of a field as **SQL Query**.
2. In the **SQL Query** section, provide the query that fetches the value from Oracle E-Business Suite and assigns the value to the field.

Create an "After Submit" Process

To allow the user to complete the transaction by clicking Submit, create an "After Submit" process.

Here is an example of an "After Submit" process in the Cancel Dispute action:

1. Create an "After Submit" process in the Processing section to update the data in Oracle E-Business Suite tables/
2. Provide the following details for the process:
    1. Identification Type: Specify "Execute Code"
    2. Source location: Specify "Local Database"
    3. Language: Specify "PL/SQL"

       PL/SQL code to cancel the dispute under PL/SQL Code section API used:

```
iex_dispute_pub.cancel_dispute
```

4. Success Message: Enter "Credit Memo Request has been cancelled successfully."

5. Error Message: Enter "Unable to process. Please try again."

This "After Submit" process will trigger the PL/SQL API code and performs the action.

Create an "After Processing" Branch

Create an "After Processing" branch in the processing section to redirect to the same page after the completion of the earlier process.

Below is an example of the creation of an "After Processing" branch in the "Cancel Dispute" form:

1. Create an After Processing branch in the processing section to redirect to the same page after completing the earlier process.

2. Select the behavior type "Page or URL (Redirect)" from the available list.

3. Select the same APEX page as the Target.

4. Set the server-side condition for "Submit Button" and "Cancellation Comment" as `dispute status <> 'CANCELLED'`. This is required not to display the two fields when the page is redirected to itself.

Oracle Enterprise Command Center Framework Push Model Integration

The Push model is required to reflect the new changes of the updated transaction in Oracle Enterprise Command Center Framework. For more information on the Push model, see: Implementing the Push Model for Loading Data, page 10-1.

To integrate with the Push model, perform the following steps:

1. Create a legacy web service reference.

2. Provide the following details for the web service:

    1. Name: `<NAME>`

    2. URL: `<EBS_PROTOCOL>://<EBS_HOST>:<EBS_MANAGED_PORT>/ecc/ir/data/&APP_SHORT_NAME./datasets/&DATASET_KEY..json?gfmString=&ECC_GFM_TOKEN.&logLevel=ERROR&languages=US&loadType=DATA_LOAD&reqId=100&source=EBS`

        In APEX, we can refer to the values at runtime using the "`&ITEM_NAME.`" convention. The value will be substituted when the web service is triggered.

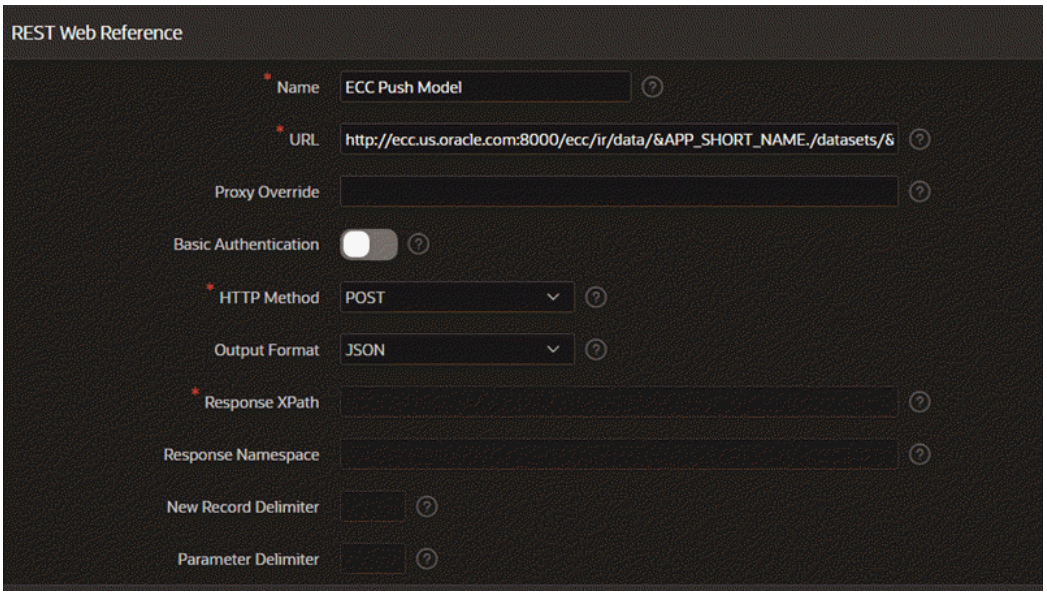
Using the Cancel Dispute example:

`&APP_SHORT_NAME.` - This will be changed to "ar" at runtime.

3. HTTP Method: JSON
4. Output Format: JSON
5. REST Input Parameters:
   - Input Type: Specified Body with Substitutions
   - Source: #PUSH_JSON_DATA#
6. Add "Content-Type" as HTTP Request Headers
7. Save the details.

An example of a legacy web service used for the Cancel Dispute example is shown below.

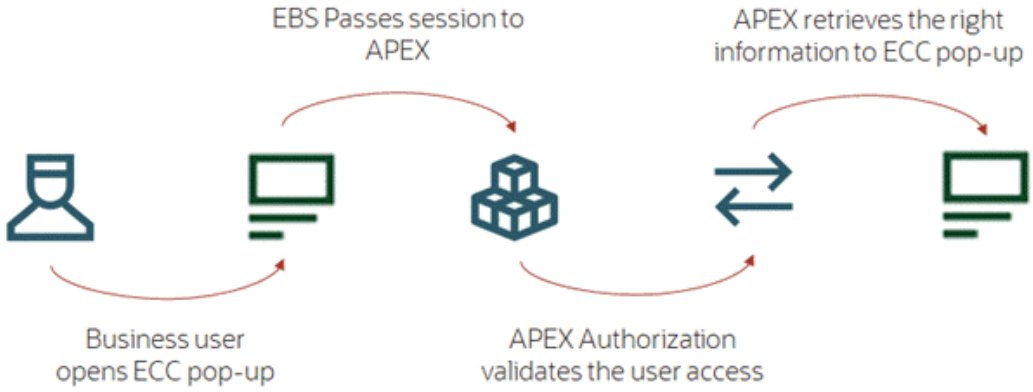
Example of a Legacy Webservice Reference



Embed in an Enterprise Command Center Pop-up Window

An APEX page can be made available from an ECC Pop-up window. The actions at runtime are described in the following illustration:

Runtime Actions for an APEX Page in an ECC Pop-up Window



1. The user opens the Oracle Enterprise Command Center pop-up window.
2. Oracle E-Business Suite passes the session to APEX.
3. APEX Authorization validates the user access.
4. APEX retrieves the appropriate information and sends it back to the ECC window.

As the authorization scheme validates both authentication and authorization of the user, the user does not need to log in again to view the APEX page. To disable the login page, enable the **Deep Linking** option under Security Attributes in Shared Components.

To embed the APEX page:

1. Set the Page Access protection as "Unrestricted" under the Security section of the APEX page as only authorized users will be able to access the page.
2. Enable the Friendly URL in APEX. Navigate to the Application home page in APEX. From the **Edit Application Properties** button, configure the **Friendly URLs** attribute.

   This will be useful while passing parameters from ECC to APEX
3. When an external page is configured as part of ECC Pop-up, cross-origin issues might arise. To resolve the cross-origin issue, set the Oracle E-Business Suite URL as CSP Frame ancestors under **HTTP Response Headers** in the APEX Application security attributes.

   To do this:

   1. Navigate to Browser Security by selecting **Application**, then **Shared Components**, then **Security Attributes**, then **Security**, and finally **Browser Security**.

2. For **HTTP Response Headers**, enter the following:

   ```
   Content-Security-Policy: frame-ancestors <EBS_PROTOTCOL>:
   //<EBS_HOST>:<EBS_PORT>;
   ```

3. Add hidden fields:

   - APP_SHORT_NAME (with a static value as that of the respective application short name in ECC),
   - DATASET_KEY (with a static value as that of the respective data set key in ECC)
   - PUSH_JSON_DATA for the page

   These fields are required while triggering the ECC Push Model to update the data.

4. Provide the following details for the Push Model process for the page. The process below will fetch the updated fields and their respective identifier (ECC_SPEC_ID) and prepare the data in JSON format.

   After that, a request is triggered to ECC using ECC Push Model Integration with JSON data. This updates the corresponding fields for the respective ECC_SPEC_ID in Oracle Enterprise Command Center Framework..

   - Name: `<PROCESS_NAME>`
   - Type: `<EXECUTE_CODE>`
   - Source Location: Local Database
   - Language: PL/SQL
   - PL/SQL Code:

```
declare
PUSH_DATA_JSON VARCHAR2(32767) :=
'{"metadata":{"columnNames":"
<COLUMN_NAMES_TO_BE_UPDATED_SEPARATED_BY_COMMA", "primarykey":"
ECC_SPEC_ID", operation":"REPLACE"},"records":[[#PUSH_DATA#]]}';

PUSH_DATA_VALUE VARCHAR2(32767);

begin

SELECT replace(PUSH_DATA, '"', 'null') INTO PUSH_DATA_VALUE FROM
(select
'<RESPECTIVE_COLUMN_VALUES_IN_DOUBLE_QUOTES_SEPERATED_BY_COMMA>')
;

-- Fetch GFM Token value
-- 60 is the value of time in seconds till which the token will
be valid

:ECC_GFM_TOKEN := fnd_gfm.one_time_use_store('ECC',60);

-- Fetch Push Data value
:PUSH_JSON_DATA := REPLACE(PUSH_DATA_JSON, '#PUSH_DATA#',
PUSH_DATA_VALUE);

END;
```

The following is an example of a Push Model Query for a Cancel Dispute transaction; this PL/SQL code pushes the updated details of the cancelled dispute:

```
declare

PUSH_DATA_JSON VARCHAR2(32767) := '{"metadata":{"columnNames":"
DISPUTE_NUMBER,ECC_SPEC_ID,DISPUTE_STATUS_CODE,ECC_LAST_UPDATE_DATE","
primarykey":"ECC_SPEC_ID","operation":"REPLACE"},"records":
[[#PUSH_DATA#]]}';
PUSH_DATA_VALUE VARCHAR2(32767);

begin

SELECT replace(PUSH_DATA, '"', 'null') INTO PUSH_DATA_VALUE FROM
(select '"' || DISPUTE_NUMBER || ECC_SPEC_ID || '","' ||
DISPUTE_STATUS_CODE || ECC_LAST_UPDATE_DATE AS PUSH_DATA

from (SELECT /*+ leading(cmreq_v.temp.process_t) full(cmreq_v.temp.
process_t) */ DISPUTE_NUMBER,

ECC_SPEC_ID,DISPUTE_STATUS_CODE, to_char(ECC_LAST_UPDATE_DATE, 'yyyy-MM-
dd"T"HH24:mi:ss"Z"') ECC_LAST_UPDATE_DATE, LANGUAGE

FROM ari_ecc_cmreq_v cmreq_v where dispute_number = :DISPUTE_NO AND
language in ('US')));


:ECC_GFM_TOKEN := fnd_gfm.one_time_use_store('ECC',60);


:PUSH_JSON_DATA := REPLACE(PUSH_DATA_JSON, '#PUSH_DATA#',
PUSH_DATA_VALUE);
END;
```

This query will trigger a "Data Load" request and update the data in Oracle Enterprise Command Center Framework.

Changes in Oracle E-Business Suite

Additional changes are required within Oracle E-Business Suite. You must define a function for the APEX page to cancel a dispute. The function definition would include the following attributes:

- Function Code: `<FUNCTION_CODE>`
- Function Name: `<Function Name>`
- HTML Call: `GWY.jsp?targetAppType=APEX&targetAPEXPage=<APEX_PAGE_URL>`

An example for the Cancel Dispute function in Oracle E-Business Suite is:

- Function Code: APEX_DISPUTE
- Function Name: APEX Dispute Details
- HTML Call: `GWY.jsp?targetAppType=APEX&targetAPEXPage=/xx_apex/r/cancel-dispute/home`

For more information on defining functions, see Overview of Function Security, *Oracle E-Business Suite Security Guide*.

Changes in Oracle Enterprise Command Center Framework

After creating the Oracle E-Business Suite function, configure the function as a link action in Oracle Enterprise Command Center Framework components, such as the Results Table, Grid, and Diagram, where the ECC Pop-up functionality is available. To do so, perform the following steps:

1. In the ECC component's configuration, enable **Display in Popup**.
2. Configure the URL with the Oracle E-Business Suite function ID along with required parameters.

   Here is an example for an APEX Cancel Dispute Page URL:

   `/OA_HTML/OA.jsp?OAFunc=APEX_DISPUTE&DISPUTE_NO={0}`
3. Select **Refresh Contents** to see the data refreshed after closing the ECC Pop-up.
4. Close the ECC Pop-up to view the updated details in the ECC Dashboard.

Data Sets and ECC_SPEC_ID Structures

Overview

This section lists data set details and ECC_SPEC_ID structure information by product.

Activity Audit (Oracle Enterprise Command Center Framework)

Data Set Key: `activity-audit`

Data Set Name

Activity Audit

Description (Purpose of the Data Set)

This application is used to track all the actions performed by a user in Enterprise Command Center dashboards. This application is accessible to the Admin user

Oracle E-Business Suite Table Names

```
ECC_USER_ACTION_AUDIT  (ECC table)
```

ECC_SPEC_ID Columns

```
AUDIT_ID
```

ECC_SPEC_ID Concatenation Structure

None.

Advanced Pricing

Data Set Key: `qp-headers`

Oracle E-Business Suite Table Names

```
QP_LIST_HEADERS_TL
QP_LIST_HEADERS_B
```

ECC_SPEC_ID Columns

```
QLB.LIST_HEADER_ID
```

ECC_SPEC_ID Concatenation Structure

```
QLB.LIST_HEADER_ID
```

Data Set Key: `qp-lines`

Oracle E-Business Suite Table Names

```
QP_LIST_LINES
QP_PRICING_ATTRIBUTES
QP_LIST_HEADERS
```

ECC_SPEC_ID Columns

```
QPL.LIST_HEADER_ID
QPL.LIST_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
QPL.LIST_HEADER_ID|| ''-''|| TQPL.LIST_LINE_ID
```

Data Set Key: `qp-qualifiers`

Oracle E-Business Suite Table Names

```
QP_QUALIFIERS
QP_ECC_STAGING_TABLE
```

ECC_SPEC_ID Columns

```
QPQ.QUALIFIER_ID
```

ECC_SPEC_ID Concatenation Structure

```
QPQ.QUALIFIER_ID
```

Asset Tracking (CSE)

Data Set Key: `oat-assets`

Data Set Name

Oracle Asset Tracking: Instances

Description (Purpose of the Data Sets)

All Active Item Instances for IB trackable items with no Fixed Asset association.

Oracle E-Business Suite Table Names

```
csi_item_instances,
csi_transactions
```

ECC_SPEC_ID Columns

```
cii.instance_id,
ct.transaction_id
```

ECC_SPEC_ID Concatenation Structure

```
(cii.instance_id || '-' || ct.transaction_id)
```

Data Set Key: `oat-capitalized-assets`

Data Set Name

Oracle Asset Tracking: Assets

Description (Purpose of the Data Set)

All Active Assets for IB Trackable items.

Oracle E-Business Suite Table Names

```
csi_i_assets
```

ECC_SPEC_ID Columns

```
cia.fa_asset_id,
cii.instance_id
```

ECC_SPEC_ID Concatenation Structure

```
(cia.fa_asset_id || '-' || cii.instance_id)
```

Assets

Data Set Key: `fa-asset`

Data Set Name

Assets

Description (Purpose of the Data Set)

Asset registration details for all corporate asset books.

Asset financial and useful asset life information to all the assets depreciated by base-life depreciation method.

Asset expense account assignments and asset location based on location key flexfield.

Oracle E-Business Suite Table Names

```
fa_additions_b ad
fa_books bk
fa_asset_invoices ai
fa_distribution_history dh
gl_code_combinations
expense_cc
```

ECC_SPEC_ID Columns

```
ad.asset_id
bk.transaction_header_id_in
NVL (ai.source_line_id, 0)
dh.distribution_id
expense_cc.code_combination_id
```

ECC_SPEC_ID Concatenation Structure

```
ad.asset_id || '-' || bk.transaction_header_id_in || '-' || NVL (ai.
source_line_id, 0) || '-' || dh.distribution_id || '-' || expense_cc.
code_combination_id
```

Data Set Key: `fa-rec`

Data Set Name

Reconciliation

Description (Purpose of the Data Set)

Reconciliation between FA accounts in SLA (Sub-Ledger Accounting) and in General Ledger for asset cost accounts and accumulated depreciation for all asset transactions except "DEPRECIATION".

Oracle E-Business Suite Table Names

```
Xla_Ae_Lines Lines
```

ECC_SPEC_ID Columns

```
FA_AEL.TRX_HDR_ID
FA_AEL.ADJUSTMENT_LINE_ID
LINES.AE_HEADER_ID
LINES.AE_LINE_NUM
Lines.GL_SL_LINK_ID
```

ECC_SPEC_ID Concatenation Structure

```
FA_AEL.TRX_HDR_ID ||'-'||FA_AEL.ADJUSTMENT_LINE_ID||'-'||LINES.
AE_HEADER_ID||'-'||LINES.AE_LINE_NUM||'-'||lines.GL_SL_LINK_ID
```

Data Set Key: `fa-masstrans`

Data Set Name

Mass Transactions

Description (Purpose of the Data Set)

Pending mass additions in all addition queues except "SPLIT" and "POSTED".

All pending mass transactions.

Oracle E-Business Suite Table Names

```
FA_MASS_TRANSFERS mt
fa_mass_revaluations mr,
FA_MASS_REVALUATION_RULES mrr
fa_mass_changes mc
FA_MASS_RECLASS mr
FA_MASS_RETIREMENTS mt
fa_mass_additions ma
```

ECC_SPEC_ID Columns

```
NVL(mt.CONCURRENT_REQUEST_ID, 0)
mt.MASS_TRANSFER_ID
NVL(mr.last_request_id,0)
mr.mass_reval_id
mrr.ASSET_ID
NVL(mc.concurrent_request_id,0)
mc.mass_change_id
NVL(mr.concurrent_request_id, 0)
mr.MASS_RECLASS_ID
NVL(mt.RETIRE_REQUEST_ID,0)
mt.MASS_RETIREMENT_ID
ma.MASS_ADDITION_ID
```

ECC_SPEC_ID Concatenation Structure

```
NVL(mt.CONCURRENT_REQUEST_ID, 0)||'-'||mt.MASS_TRANSFER_ID
NVL(mr.last_request_id,0)||'-'||mr.mass_reval_id||'-'||mrr.ASSET_ID
NVL(mc.concurrent_request_id,0)||'-'||mc.mass_change_id
NVL(mr.concurrent_request_id, 0)||'-'||mr.MASS_RECLASS_ID
NVL(mt.RETIRE_REQUEST_ID,0)||'-'||mt.MASS_RETIREMENT_ID
ma.MASS_ADDITION_ID || ''
```

Data Set Key: `fa-clr`

Data Set Name

Pipeline

Description (Purpose of the Data Set)

All invoices in Payables that are not yet transferred to FA and marked as "Track As Assets".

All Projects and tasks in Projects and not transferred yet to FA.

Oracle E-Business Suite Table Names

```
ap_invoices_all inv
ap_invoice_lines_all invl
pa_project_asset_lines_all pal
pa_project_assets_all paa
```

ECC_SPEC_ID Columns

```
inv.invoice_id
invl.line_number
invd.invoice_distribution_id
paa.project_id
paa.project_asset_id
pal.project_asset_line_id
pal.project_asset_line_detail_id
```

ECC_SPEC_ID Concatenation Structure

```
'INV'|| '-'|| inv.invoice_id|| '-'|| invl.line_number||'-'|| invd.
invoice_distribution_id
'PROJ'|| '-'|| paa.project_id|| '-'|| paa.project_asset_id|| '-'|| pal.
project_asset_line_id|| '-'|| pal.project_asset_line_detail_id
```

Cash Management

Data Set Key: `ce-bank-statement`

Data Set Name

Bank Statements

Description (Purpose of the Data Set)

All open bank statements details.

Oracle E-Business Suite Table Names

```
CE_BANK_ACCOUNTS
ce_statement_headers
ce_statement_lines
```

ECC_SPEC_ID Columns

```
cebs.bank_account_num
cebs.statement_header_id
cebs.statement_line_id
```

ECC_SPEC_ID Concatenation Structure

```
cebs.bank_account_num||''-''||
cebs.statement_header_id||''-''||
cebs.statement_line_id
```

Data Set Key: `ce-recon-available`

Data Set Name

Bank Available Transactions

Description (Purpose of the Data Set)

Available transactions from AP, AR, GL, Payroll, Treasury, and cash flow statements.

Oracle E-Business Suite Table Names

```
gl_je_lines
gl_je_headers
ce_bank_accounts
ap_checks_all
ar_cash_receipts_all
ar_cash_receipt_history_all
ce_statement_lines
ce_statement_headers
pay_assignment_actions
pay_org_payment_methods_f
xtr_settlement_summary
```

ECC_SPEC_ID Columns

```
TRX_ID
CURRENCY_CODE
CASH_RECEIPT_ID
BANK_ACCOUNT_ID
```

ECC_SPEC_ID Concatenation Structure

```
TRX_ID||''-''||CURRENCY_CODE||''-''||
CASH_RECEIPT_ID||''-''||
BANK_ACCOUNT_ID
```

Channel Revenue Management

Data Set Key: `ozf-open-claim`

Data Set Name

Open Claims

Description (Purpose of the Data Set)

This data set holds information about channel revenue management details for the open claim dashboard.

Oracle E-Business Suite Table Names

```
OZF_CLAIMS_ALL C
```

ECC_SPEC_ID Columns

```
C.CLAIM_ID
```

ECC_SPEC_ID Concatenation Structure

```
C.CLAIM_ID ||'-' || null ||'-' || null
```

Data Set Key: `ozf-closed-claim`

Data Set Name

Closed Claims

Description (Purpose of the Data Set)

This data set holds information about channel revenue management details for the closed claim dashboard.

Oracle E-Business Suite Table Names

```
OZF_CLAIMS_ALL C
OZF_CLAIM_LINES_ALL L
OZF_FUNDS_UTILIZED_ALL_B FU
QP_LIST_HEADERS_B QP
```

ECC_SPEC_ID Columns

```
C.CLAIM_ID
QP.LIST_HEADER_ID
FU.PRODUCT_ID
L.ITEM_ID
```

ECC_SPEC_ID Concatenation Structure

```
C.CLAIM_ID ||'-'|| QP.LIST_HEADER_ID ||'-' || NVL(FU.PRODUCT_ID,L.
ITEM_ID)
```

Data Set Key: `ozf-budget`

Data Set Name

Budgets

Description (Purpose of the Data Set)

This data set holds information about channel revenue management details for the budget dashboard.

Oracle E-Business Suite Table Names

```
OZF_FUNDS_UTILIZED_ALL_B
OZF_FUNDS_ALL_B
```

ECC_SPEC_ID Columns

```
FUND_ID
PLAN_TYPE
PLAN_ID
BILLTO_CUST_ACCOUNT_ID
CUST_ACCOUNT_ID
BILL_TO_SITE_USE_ID
SHIP_TO_SITE_USE_ID
PRODUCT_LEVEL_TYPE
PRODUCT_ID
ACTIVITY_DATE
```

ECC_SPEC_ID Concatenation Structure

```
fund_id || '-' || plan_type || '-' || plan_id || '-' ||
billto_cust_account_id || '-' ||
cust_account_id || '-' || bill_to_site_use_id || '-' ||
ship_to_site_use_id || '-' ||
product_level_type || '-' || product_id || '-' || activity_date || '-'
|| type_flag
```

Contract Lifecycle Management for Public Sector

`po-clm-requisitions`

Data Set Name

Requisitions

Description (Purpose of the Data Set)

This enables contracting officers to track requisitions and workload assignments.

It also provides the capability to review, monitor, alert and analyze requisitions for both contracting officers and program management office users.

Oracle E-Business Suite Table Names

```
PO_REQUISITION_HEADERS_ALL PRH
PO_REQUISITION_LINES_ALL PRL
PO_REQ_DISTRIBUTIONS_ALL PRD
PO_DRAFTS PD
PO_CLM_ASSIGNMENTS PCA
PO_CLM_ASSGNMT_MILESTONE_DTLS PMD
```

ECC_SPEC_ID Columns

```
PRH.REQUISITION_HEADER_ID
PRL.REQUISITION_LINE_ID
PRD.DISTRIBUTION_ID
PCA.ASSIGNMENT_ID
PMD.MILESTONE_CODE
PMD.MILESTONE_TYPE
PRH.PAR_DRAFT_ID
PRH.PAR_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

1.
```
REQUISITION_HEADER_ID
|| '-'
|| REQUISITION_LINE_ID
|| '-'
|| DISTRIBUTION_ID
|| '-'
|| ASSIGNMENT_ID
|| '-'
|| MILESTONE_CODE
|| '-'
|| MILESTONE_TYPE
|| '-'
|| PAR_DRAFT_ID
|| '-'
|| PAR_LINE_ID ECC_SPEC_ID
```

2.
```
ASSIGNMENT_ID
|| '-'
|| MILESTONE_CODE
|| '-'
|| MILESTONE_TYPE ECC_SPEC_ID
```

Data Set Key: `po-clm-award`

Data Set Name

Awards

Description (Purpose of the Data Set)

This data set provides information on awards and their modifications, related compliance information such as Contract Action Report (CAR), Federal Procurement Data System - Next Generation (FPDS-NG), and Justification and Approval (J&A) postings.

Oracle E-Business Suite Table Names

```
PO_HEADERS_ALL
PO_DRAFTS
PO_LINES_ALL
PO_LINE_LOCATIONS_ALL
PO_DISTRIBUTIONS_ALL
```

ECC_SPEC_ID Columns

```
PO_HEADER_ID
DRAFT_ID
PO_LINE_ID
LINE_LOCATION_ID
PO_DISTRIBUTION_ID
```

ECC_SPEC_ID Concatenation Structure

```
PO_HEADER_ID || '-' || DRAFT_ID || '-' || PO_LINE_ID || '-' ||
LINE_LOCATION_ID|| '-' || PO_DISTRIBUTION_ID
```

Data Set Key: `po-clm-idv`

Data Set Name

IDVs

Description (Purpose of the Data Set)

This data set provides complete information on Indefinite Definite Vehicles (IDVs).

Oracle E-Business Suite Table Names

```
PO_HEADERS_ALL
PO_DRAFTS
PO_LINES_ALL
```

ECC_SPEC_ID Columns

```
PO_HEADER_ID
DRAFT_ID
PO_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
PO_HEADER_ID || '-' || DRAFT_ID || '-' || PO_LINE_ID
```

Data Set Key: `po-clm-par`

Data Set Name

Post Award Requests

Description (Purpose of the Data Set)

This data set provides information on Post Award Requests (PARs)

Oracle E-Business Suite Table Names

```
PO_HEADERS_ALL
PO_DRAFTS
PO_LINES_ALL
PO_LINE_LOCATIONS_ALL
PO_DISTRIBUTIONS_ALL
```

ECC_SPEC_ID Columns

```
PO_HEADER_ID
DRAFT_ID
PO_LINE_ID
LINE_LOCATION_ID
PO_DISTRIBUTION_ID
```

ECC_SPEC_ID Concatenation Structure

```
PO_HEADER_ID || '-' || DRAFT_ID || '-' || PO_LINE_ID || '-' ||
LINE_LOCATION_ID|| '-' || PO_DISTRIBUTION_ID
```

Data Set Key: `po-clm-deliverables`

Data Set Name

CLM Deliverables

Description (Purpose of the Data Set)

Information about deliverables and their statuses.

Oracle E-Business Suite Table Names

```
OKC_DELIVERABLES
PO_HEADERS_ALL
```

ECC_SPEC_ID Columns

```
DELIVERABLE_ID
PO_HEADER_ID
```

ECC_SPEC_ID Concatenation Structure

```
    DELIVERABLE_ID || '-' || PO_HEADER_ID
```

Data Set Key: `po-clm-protest`

Data Set Name

CLM Protest

Description (Purpose of the Data Set)

This data set provides information about protests created and statuses.

Oracle E-Business Suite Table Names

```
PO_PROTESTS
PO_HEADERS_ALL
```

ECC_SPEC_ID Columns

```
PROTEST_ID
PO_HEADER_ID
```

ECC_SPEC_ID Concatenation Structure

```
PROTEST_ID || '-' || PO_HEADER_ID
```

Data Set Key: `po-clm-closeout`

Data Set Name

CLM Closeout Tasks

Description (Purpose of the Data Set)

This data set provides information on closeout tasks and their statuses.

Oracle E-Business Suite Table Names

```
OKC_DELIVERABLES
PO_HEADERS_ALL
```

ECC_SPEC_ID Columns

```
DELIVERABLE_ID
PO_HEADER_ID
```

ECC_SPEC_ID Concatenation Structure

```
DELIVERABLE_ID || '-' || PO_HEADER_ID
```

Data Set Key: `po-clm-solicitations`

Data Set Name

Solicitations

Description (Purpose of the Data Set)

This data set provides information on solicitations.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all pah
pon_auction_item_prices_all paip
```

ECC_SPEC_ID Columns

```
pah.auction_header_id
paip.line_number
```

ECC_SPEC_ID Concatenation Structure

```
pah.auction_header_id || '-'
|| paip.line_number ecc_spec_id
```

Data Set Key: `po-clm-sol-time`

Data Set Name

Solicitation Time Line

Description (Purpose of the Data Set)

This data set provides information on time taken for each stage of solicitation.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all pah
```

ECC_SPEC_ID Columns

```
pah.auction_header_id
```

ECC_SPEC_ID Concatenation Structure

```
auction_header_id||''-''||duration_type ECC_SPEC_ID
```

Data Set Key: `po-clm-sol-protests`

Data Set Name

Solicitation Protests

Description (Purpose of the Data Set)

This provides information on protests created as part of solicitation process.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all pah
po_protests prt
```

ECC_SPEC_ID Columns

```
pah.auction_header_id
prt.protest_id
```

ECC_SPEC_ID Concatenation Structure

```
pah.auction_header_id||'-'|| prt.protest_id ECC_SPEC_ID
```

Data Set Key: `po-clm-sol-offers`

Data Set Name

Solicitation Responses

Description (Purpose of the Data Set)

This provides information on responses to the solicitations that are published.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all pah
pon_bid_headers pbh
pon_bid_item_prices pbip
```

ECC_SPEC_ID Columns

```
pah.auction_header_id
pbh.bid_number
pbip.line_number
```

ECC_SPEC_ID Concatenation Structure

```
pah.auction_header_id
|| '-'
|| pbh.bid_number
|| '-'
|| pbip.line_number ecc_spec_id
```

Data Set Key: `po-clm-sol-forms`

Data Set Name

Solicitation Forms

Description (Purpose of the Data Set)

This will provide information about the forms such as pre-solicitation forms, Small Business Coordination Record (SBCR), and FedBizOps (FBO), that are used as part of solicitation process.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all ah,
PON_FORMS_SECTIONS Forms
PON_FORMS_INSTANCES pfi
okc_rep_contracts_all orca
```

ECC_SPEC_ID Columns

```
ah.auction_header_id
pfi.form_id
Forms.form_code
orca.contract_number
```

ECC_SPEC_ID Concatenation Structure

1. `ah.auction_header_id||'-'||pfi.form_id||'-'||'FEDBIZ'||'-'||Forms.form_code ECC_SPEC_ID`

2. `ah.auction_header_id|'-'||'SBCR'||'-'||orca.contract_number ECC_SPEC_ID`

Data Set Key: `po-clm-sol-deliverables`

Data Set Name

Solicitation Deliverables

Description (Purpose of the Data Set)

This data set provides information exclusively on deliverables created as part of solicitation document.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all pah
okc_deliverables deliverable
```

ECC_SPEC_ID Columns

```
pah.auction_header_id
deliverable.deliverable_id
```

ECC_SPEC_ID Concatenation Structure

```
pah.auction_header_id||''-''||deliverable.deliverable_id as ECC_SPEC_ID
```

Data Set Key: `po-clm-acquisitions`

Data Set Name

Acquisitions

Description (Purpose of the Data Set)

This data set provides information on an acquisition plan and details of each milestone.

Oracle E-Business Suite Table Names

```
okc_rep_contracts_all orc
okc_deliverables ord
```

ECC_SPEC_ID Columns

```
orc.contract_id
ord.deliverable_id
```

ECC_SPEC_ID Concatenation Structure

```
orc.contract_id||"-"||ord.deliverable_id as ECC_SPEC_ID
```

Cost Management

Data Set Key: `cst-phc-invt`

Data Set Name

Inventory Transactions

Description (Purpose of the Data Set)

This data set holds the details of Inventory Transactions for the Period Health Check dashboard.

Oracle E-Business Suite Table Names

```
mtl_material_transactions (MMT)
mtl_transactions_interface (MTI)
mtl_material_transactions_temp (MMTT)
WSH_DELIVERY_DETAILS (WSH)
CST_LC_ADJ_INTERFACE (LCM)
gmf_lc_adj_headers_v (glah)
gmf_lc_adj_transactions (glat)
gmf_period_balances (GPB)
gmf_lot_cost_adjustments (lca)
cm_adjs_dtl (cad)
gl_aloc_dtl (dtl)
GMF_INVOICE_DISTRIBUTIONS (GID)
```

ECC_SPEC_ID Columns

```
MMT.rowid
MTI.rowid
MMTT.Rowid
WSH.ROW_ID
LCM.ROWID
GLAT.ROWID
GLAT.ship_header_id
GLAT.ship_line_id
GLAT.adj_transaction_id
GPB.PERIOD_BALANCE_ID
GPB.INVENTORY_ITEM_ID
GPB.ROWID
LCA.ADJUSTMENT_ID
LCA.INVENTORY_ITEM_ID
LCA.ROWID
CAD.COST_ADJUST_ID
CAD.ROWID
DTL.ALLOC_ID
MSI.INVENTORY_ITEM_ID
DTL.LINE_NO
DTL.ALLOCDTL_ID
DTL.ROWID
gid.rowid
```

ECC_SPEC_ID Concatenation Structure

```
'MMT' || MMT.rowid
UNION ALL
'MMT' || MMT.rowid
UNION ALL
'MTI' || MTI.rowid
UNION ALL
'MMTT' || MMTT.Rowid
UNION ALL
'WSH' || WSH.ROW_ID
UNION ALL
'LCM' || LCM.ROWID
UNION ALL
'LCM' || RT.ROWID || GLAH.ship_header_id || GLAH.ship_line_id|| glah.
adjustment_num||glah.component_type|| glah.cost_cmpntcls_id || glah.
cost_analysis_code
UNION ALL
'LCM' || GLAT.ROWID || GLAT.ship_header_id || GLAT.ship_line_id || GLAT.
adj_transaction_id
UNION ALL
'GPB' || GPB.PERIOD_BALANCE_ID ||'-' ||GPB.INVENTORY_ITEM_ID || GPB.
ROWID
UNION ALL
'LCA' ||LCA.ADJUSTMENT_ID ||'-' ||LCA.INVENTORY_ITEM_ID ||LCA.ROWID
UNION ALL
'CAD' || CAD.COST_ADJUST_ID ||CAD.ROWID
UNION ALL
'GLEXP' || DTL.ALLOC_ID || MSI.INVENTORY_ITEM_ID || DTL.LINE_NO || DTL.
ALLOCDTL_ID || DTL.ROWID
UNION ALL
'GID' || gid.rowid
```

Data Set Key: `st-phc-rcv`

Data Set Name

Receiving

Description (Purpose of the Data Set)

This data set holds the details of Receiving transactions for the Period Health Check dashboard.

Oracle E-Business Suite Table Names

```
rcv_transactions_interface (rti)
gmf_rcv_accounting_txns (grat)
```

ECC_SPEC_ID Columns

```
RTI.rowid
GRAT.rowid
GRAT.rowid
```

ECC_SPEC_ID Concatenation Structure

```
'RTI' || RTI.rowid
UNION ALL
'GRAT' || GRAT.rowid
UNION ALL
'GRAT' || GRAT.rowid
```

Data Set Key: `cst-phc-gmfmfg`

Data Set Name

Manufacturing (Process)

Description (Purpose of the Data Set)

This data set holds the details of manufacturing transactions across Process Manufacturing organizations for the Period Health Check dashboard.

Oracle E-Business Suite Table Names

```
gme_resource_txns (rt)
```

ECC_SPEC_ID Columns

```
RT.rowid
```

ECC_SPEC_ID Concatenation Structure

```
'RSRC'|| RT.ROWID
```

Data Set Key: `cst-phc-mfg-dis`

Data Set Name

Manufacturing (Discrete)

Description (Purpose of the Data Set)

This data set holds the details of manufacturing transactions across Discrete Manufacturing organizations for the Period Health Check dashboard.

Oracle E-Business Suite Table Names

```
WIP_COST_TXN_INTERFACE (wcti)
WIP_MOVE_TXN_INTERFACE (wmti)
WSM_LOT_MOVE_TXN_INTERFACE (wlmti)
WSM_LOT_SPLIT_MERGES_INTERFACE (wlsmi)
WSM_SPLIT_MERGE_TXN_INTERFACE (wsmti)
```

ECC_SPEC_ID Columns

```
WCTI.ROWID
WMTI.ROWID
wlmti.rowid
wlsmi.rowid
wsmti.rowid
```

ECC_SPEC_ID Concatenation Structure

```
'WCTI' || WCTI.ROWID
UNION ALL
'WMTI' || WMTI.ROWID
UNION ALL
'WSMLMIT' || wlmti.rowid
UNION ALL
'WSMLSMTI'|| wlsmi.rowid
UNION ALL
'WSMSMTI'|| wsmti.rowid
```

Data Set Key: `cst-phc-eam`

Data Set Name

Maintenance

Description (Purpose of the Data Set)

This data set holds the details of maintenance work orders for the Period Health Check dashboard.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
```

ECC_SPEC_ID Columns

```
WDJ.rowid
```

ECC_SPEC_ID Concatenation Structure

```
'EAM' || WDJ.ROWID
```

Data Set Key: `cst-phc-ovw`

Data Set Name

Overview

Description (Purpose of the Data Set)

This data set holds the Overview details for the Period Health Check dashboard.

Oracle E-Business Suite Table Names

```
mtl_material_transactions (MMT)
mtl_transactions_interface (MTI)
mtl_material_transactions_temp (MMTT)
WSH_DELIVERY_DETAILS (WSH)
CST_LC_ADJ_INTERFACE (LCM)
gmf_lc_adj_headers_v (glah)
gmf_lc_adj_transactions (glat)
gmf_period_balances (GPB)
gmf_lot_cost_adjustments (lca)
cm_adjs_dtl (cad)
gl_aloc_dtl (dtl)
GMF_INVOICE_DISTRIBUTIONS (GID)
rcv_transactions_interface (rti)
gmf_rcv_accounting_txns (grat)
gme_resource_txns (rt)
WIP_COST_TXN_INTERFACE (wcti)
WIP_MOVE_TXN_INTERFACE (wmti)
WSM_LOT_MOVE_TXN_INTERFACE (wlmti)
WSM_LOT_SPLIT_MERGES_INTERFACE (wlsmi)
WSM_SPLIT_MERGE_TXN_INTERFACE (wsmti)
wip_discrete_jobs
```

ECC_SPEC_ID Columns

```
MMT.rowid
MMT.rowid
MTI.rowid
MMTT.Rowid
WSH.ROW_ID
LCM.ROWID
GLAT.ROWID
GLAT.ship_header_id
GLAT.ship_line_id
GLAT.adj_transaction_id
GPB.PERIOD_BALANCE_ID
GPB.INVENTORY_ITEM_ID
GPB.ROWID
LCA.ADJUSTMENT_ID
LCA.INVENTORY_ITEM_ID
LCA.ROWID
CAD.COST_ADJUST_ID
CAD.ROWID
DTL.ALLOC_ID
MSI.INVENTORY_ITEM_ID
DTL.LINE_NO
DTL.ALLOCDTL_ID
DTL.ROWID
gid.rowid
RTI.rowid
GRAT.rowid
GRAT.rowid
RT.rowid
WCTI.ROWID
WMTI.ROWID
wlmti.rowid
wlsmi.rowid
wsmti.rowid
WDJ.rowid
```

ECC_SPEC_ID Concatenation Structure

```
'MMT' || MMT.rowid
UNION ALL
'MMT' || MMT.rowid
UNION ALL
'MTI' || MTI.rowid
UNION ALL
'MMTT' || MMTT.Rowid
UNION ALL
'WSH' || WSH.ROW_ID
UNION ALL
'LCM' || LCM.ROWID
UNION ALL
'LCM' || RT.ROWID || GLAH.ship_header_id || GLAH.ship_line_id|| glah.
adjustment_num||glah.component_type|| glah.cost_cmpntcls_id || glah.
cost_analysis_code
UNION ALL
'LCM' || GLAT.ROWID || GLAT.ship_header_id || GLAT.ship_line_id || GLAT.
adj_transaction_id
UNION ALL
'GPB' || GPB.PERIOD_BALANCE_ID ||'-' ||GPB.INVENTORY_ITEM_ID || GPB.
ROWID
UNION ALL
'LCA' ||LCA.ADJUSTMENT_ID ||'-' ||LCA.INVENTORY_ITEM_ID ||LCA.ROWID
UNION ALL
'CAD' || CAD.COST_ADJUST_ID ||CAD.ROWID
UNION ALL
'GLEXP' || DTL.ALLOC_ID || MSI.INVENTORY_ITEM_ID || DTL.LINE_NO || DTL.
ALLOCDTL_ID || DTL.ROWID
UNION ALL
'GID' || gid.rowid
UNION ALL
'RTI' || RTI.rowid
UNION ALL
'GRAT' || GRAT.rowid
UNION ALL
'GRAT' || GRAT.rowid
UNION ALL
'RSRC'|| RT.ROWID
UNION ALL
'WCTI' || WCTI.ROWID
UNION ALL
'WMTI' || WMTI.ROWID
UNION ALL
'WSMLMIT' || wlmti.rowid
UNION ALL
'WSMLSMTI'|| wlsmi.rowid
UNION ALL
'WSMSMTI'|| wsmti.rowid
UNION ALL
'EAM' || WDJ.ROWID
```

Data Set Key: `cst-pval-batch`

Data Set Name

WIP Valuation (Process)

Description (Purpose of the Data Set)

This Data set holds the WIP Valuation (Process) data

Oracle E-Business Suite Table Names

```
CST_ECC_VAL_BATCH_SMRY_V
```

ECC_SPEC_ID Columns

```
acct_period_id,
batch_id,
report_mode_code,
txn_period_id,
code_combination_id,
usage_ind,
cost_cmpntcls_id,
cost_analysis_code,
usage_ind_alt,
final_accounted,
cost_type_id
```

ECC_SPEC_ID Concatenation Structure

```
wipval.acct_period_id
|| '-'
|| wipval.batch_id
|| '-'
|| wipval.report_mode_code
|| '-'
|| wipval.txn_period_id
|| '-'
|| wipval.code_combination_id
|| '-'
|| wipval.usage_ind
|| '-'
|| wipval.cost_cmpntcls_id
|| '-'
|| wipval.cost_analysis_code
|| '-'
|| wipval.usage_ind_alt
|| '-'
|| wipval.final_accounted
|| '-'
|| wipval.cost_type_id
```

Depot Repair

Data Set Key: `csd-repairs`

Data Set Name

Depot Repair Service Orders

Description (Purpose of the Data Set)

Service Orders that are in progress.

Oracle E-Business Suite Table Names

```
csd_repairs (cr)
cs_incidents_all_b (sr)
fnd_user
hz_parties
hz_cust_accounts
csd_flow_statuses_b
csd_flow_statuses_tl
csd_repair_types_b
csi_item_instances
cs_incident_types_tl
cs_incident_severities_tl
cs_incident_urgencies_tl
cs_incident_statuses_tl
cs_incidents_all_tl
hz_party_site_use
hz_party_sites
hz_locations
fnd_territories_tl
okc_k_headers_all_b
okc_k_lines_b
csd_repair_estimate
csd_repair_actuals
cs_hz_sr_contact_points
hr_organization_information
csd_return_types_tl
jtf_rs_resource_extns
csd_ro_service_codes
csd_ro_diagnostic_codes
csd_diagnostic_codes_b
csd_diagnostic_codes_tl
csd_service_codes_b
csd_service_codes_tl
```

ECC_SPEC_ID Columns

```
cr.repair_line_id, sr.incident_id
```

ECC_SPEC_ID Concatenation Structure

```
 'CSD-RO-' || cr.repair_line_id || '-' || sr.incident_id
```

Data Set Key: `csd-repairs-closed`

Data Set Name

Depot Resolved Service Orders

Description (Purpose of the Data Set)

Service Orders that have been resolved.

Oracle E-Business Suite Table Names

```
csd_repairs (cr)
cs_incidents_all_b (sr)
csd_repair_history (hist)
fnd_user
hz_parties
hz_cust_accounts
csd_flow_statuses_b
csi_item_instances
hz_party_site_uses
hz_party_sites
hz_parties
hz_cust_accounts
hz_locations
okc_k_headers_all_b
okc_k_lines_b
csd_repair_estimate
csd_repair_actuals
cs_hz_sr_contact_points
hr_organization_information
jtf_rs_resource_extns
csd_repair_types_tl
cs_estimate_details
csd_product_transactions
oe_order_headers_all
oe_order_lines_all
rcv_transactions
mtl_parameters
ar_lookups
wip_period_balances
csd_repair_job_xref
wip_discrete_jobs
csd_repair_actual_lines
fnd_lookup_values
csd_ro_defect_codes
mtl_system_items_kfv
wsh_delivery_details
wsh_delivery_assignments
wsh_new_deliveries
mtl_parameters
wip_entities
csd_flow_statuses_tl
cs_incident_urgencies_tl
cs_incident_types_tl
cs_incident_severities_tl
cs_incident_statuses_tl
cs_incidents_all_tl
hr_all_organization_units_tl
jtf_rs_resource_extns_tl
jtf_rs_groups_tl
hr_all_organization_units_tl
fnd_territories_tl
```

ECC_SPEC_ID Columns

```
cr.repair_line_id, sr.incident_id, hist.paramc2
```

ECC_SPEC_ID Concatenation Structure

```
'CSD-RO-'||cr.repair_line_id ||'-' ||sr.incident_id ||'-' || hist.
paramc2
```

Discrete Manufacturing

Data Set Key: `wip-wodetails`

Data Set Name

Work Orders

Description (Purpose of the Data Set)

This data set provides information on all Discrete Job Work Orders and statuses.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
mtl_reservations
oe_order_lines_all
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wdj.organization_id
mr.supply_source_header_id
ool1.line_id
```

ECC_SPEC_ID Concatenation Structure

```
(wdj.wip_entity_id||''-'' ||wdj.organization_id||''-'' ||mr.
supply_source_header_id||''-''||ool1.line_id ||''-JOB'')
```

Data Set Key: `wip-opdetails`

Data Set Name

Operations

Description (Purpose of the Data Set)

This data set provides information of each work order operations details and scheduling details.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_operations
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wo.operation_seq_num
wdj.organization_id
```

ECC_SPEC_ID Concatenation Structure

```
(wdj.wip_entity_id ||''-'' ||wo.operation_seq_num ||''-''
||wdj.organization_id ||''-OP'')
```

Data Set Key: `wip-serialdetails`

Data Set Name

Serial Numbers

Description (Purpose of the Data Set)

This data set provides information of Assembly Serials.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
mtl_serial_numbers
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wdj.organization_id
msn.serial_number
```

ECC_SPEC_ID Concatenation Structure

```
(wdj.wip_entity_id ||''-''||wdj.organization_id ||''-''||msn.
serial_number||''-SER'')
```

Data Set Key: `wip-compdetails`

Data Set Name

Components

Description (Purpose of the Data Set)

This data set provides information of components of all work orders.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_requirement_operations
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wro.operation_seq_num
wro.inventory_item_id
wdj.organization_id
```

ECC_SPEC_ID Concatenation Structure

```
(wdj.wip_entity_id||''-''||wro.operation_seq_num||''-''||wro.
inventory_item_id ||''-''||wdj.organization_id ||''-COMP'')
```

Data Set Key: `wip-opqualitydetails`

Data Set Name

Move Transactions

Description (Purpose of the Data Set)

This data set provides information about move transactions of each operation of work order.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_operations
wip_move_transactions
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wdj.organization_id
wo.operation_seq_num
wmt.transaction_id
```

ECC_SPEC_ID Concatenation Structure

```
(wdj.wip_entity_id ||''-''||wdj.organization_id ||''-''||wo.
operation_seq_num||''-''||wmt.transaction_id||''-TXN'')
```

Data Set Key: `wip-ospdetails`

Data Set Name

Resources

Description (Purpose of the Data Set)

This data set provides information of Resources and resource scheduling details.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_operations
bom_resources
po_distributions_all
po_releases_all
po_requisition_lines_all
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wo.operation_seq_num
br.resource_code
wdj.organization_id
prl.requisition_header_id
prl.requisition_line_id
UNION
wdj.wip_entity_id
wo.operation_seq_num
br.resource_code
wdj.organization_id
pd.po_header_id
pd.po_line_id
pr.po_release_id
UNION
wdj.wip_entity_id
wo.operation_seq_num
br.resource_code
wdj.organization_id
pd.po_header_id
pd.po_line_id
```

ECC_SPEC_ID Concatenation Structure

```
wdj.wip_entity_id||''-''||wo.operation_seq_num||''-''||br.
resource_code||''-''||wdj.organization_id||''-''||prl.
requisition_header_id ||''-''||prl.requisition_line_id||''-OSP''
UNION
wdj.wip_entity_id||''-''||wo.operation_seq_num||''-''||br.
resource_code||''-''||wdj.organization_id||''-''||pd.po_header_id
||''-''||pd.po_line_id|| ''-''||pr.po_release_id||''-OSP''
UNION
wdj.wip_entity_id||''-''||wo.operation_seq_num||''-''||br.
resource_code||''-''||wdj.organization_id||''-''||pd.po_header_id
||''-''||pd.po_line_id||''-OSP''
```

Data Set Key: `wip-ospdetails2`

Data Set Name

Outside Processing

Description (Purpose of the Data Set)

This data set provides information about Outside Processing jobs.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_operations
bom_resources
po_distributions_all
po_releases_allpo_requisition_lines_all
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wo.operation_seq_num
br.resource_code
wdj.organization_id
prl.requisition_header_id
prl.requisition_line_id
UNION
wdj.wip_entity_id
wo.operation_seq_num
br.resource_code
wdj.organization_id
pd.po_header_id
pd.po_line_id
pr.po_release_id
UNION
wdj.wip_entity_id
wo.operation_seq_num
br.resource_code
wdj.organization_id
pd.po_header_id
pd.po_line_id
```

ECC_SPEC_ID Concatenation Structure

```
wdj.wip_entity_id||''-''||wo.operation_seq_num||''-''||br.
resource_code||''-''||wdj.organization_id||''-''||prl.
requisition_header_id ||''-''||prl.requisition_line_id||''-OSP''
UNION
wdj.wip_entity_id||''-''||wo.operation_seq_num||''-''||br.
resource_code||''-''||wdj.organization_id||''-''||pd.po_header_id
||''-''||pd.po_line_id|| ''-''||pr.po_release_id||''-OSP''
UNION
wdj.wip_entity_id||''-''||wo.operation_seq_num||''-''||br.
resource_code||''-''||wdj.organization_id||''-''||pd.po_header_id
||''-''||pd.po_line_id||''-OSP''
```

Discrete Quality

Data Set Key: `qa-quality`

Data Set Name

Collection Results

Description (Purpose of the Data Set)

All the Quality results collected across plans and published to display on the Enterprise Command Center dashboard.

Oracle E-Business Suite Table Names

```
QA_ECC_RESULTS_TAB (ECC Table)
QA_ECC_RESULTS_V (ECC View)
```

ECC_SPEC_ID Columns

```
ECC_SPEC_ID
```

ECC_SPEC_ID Concatenation Structure

```
plan_id-collection_id-occurrence
```

Data Set Key: `qa-nc`

Data Set Name

Nonconformances

Description (Purpose of the Data Set)

All the Quality results collected across NC plans and published to display on the Enterprise Command Center dashboard.

Oracle E-Business Suite Table Names

```
QA_ECC_RESULTS_TAB (ECC Table)
QA_ECC_RESULTS_NC_V (ECC View)
```

ECC_SPEC_ID Columns

```
ECC_SPEC_ID
```

ECC_SPEC_ID Concatenation Structure

```
plan_id-collection_id-occurrence
```

Data Set Key: `qa-ca`

Data Set Name

Corrective Actions

Description (Purpose of the Data Set)

All the Quality results collected across CA plans and published to display on the Enterprise Command Center dashboard.

Oracle E-Business Suite Table Names

```
QA_ECC_RESULTS_TAB (ECC Table)
QA_ECC_RESULTS_CA_V (ECC View)
```

ECC_SPEC_ID Columns

```
ECC_SPEC_ID
```

ECC_SPEC_ID Concatenation Structure

```
plan_id-collection_id-occurrence
```

Enterprise Asset Management

Data Set Key: `eam-wo`

Data Set Name

Work Orders (Maintenance)

Description (Purpose of the Data Set)

Open Work Order information.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs,
wip_operations,
wip_operation_resources,
wip_op_resource_instances
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id,
wo.operation_seq_num,
wor.resource_seq_num,
wor.resource_id,
wori.instance_id,
wori.serial_number
```

ECC_SPEC_ID Concatenation Structure

```
( wdj.wip_entity_id || '-' ||
wo.operation_seq_num || '-' ||
wor.resource_seq_num || '-' ||
wor.resource_id || '-INSTANCE-' ||
wori.instance_id || '-EQUIPMENT-' ||
wori.serial_number
```

Data Set Key: `eam-wr`

Data Set Name

Work Requests (Maintenance)

Description (Purpose of the Data Set)

Contains work request information for Work Orders dashboard.

Oracle E-Business Suite Table Names

```
wip_eam_work_requests
```

ECC_SPEC_ID Columns

```
work_request_number
```

ECC_SPEC_ID Concatenation Structure

None.

Data Set Key: `eam-store-mat`

Data Set Name

Stores Materials (Maintenance)

Description (Purpose of the Data Set)

Material information along with information like on-hand quantity, available quantity, manufacturer, and cross-reference information.

Oracle E-Business Suite Table Names

```
mtl_system_items_kfv,
mtl_cross_references_b,
mtl_mfg_part_numbers,
mtl_secondary_inventories
mtl_system_items_kfv
```

ECC_SPEC_ID Columns

```
msi.inventory_item_id,
msi.organization_id,
mcr.cross_reference_id,
mpn.manufacturer_id,
mpn.mfg_part_num,
msinv.secondary_inventory_name,
msi.concatenated_segments (asset_bom)
```

ECC_SPEC_ID Concatenation Structure

```
( msi.inventory_item_id ||'-' ||
msi.organization_id ||'-CROSS_REF-' ||
mcr.cross_reference_id ||'-MANUFACTURER-' ||
mpn.manufacturer_id ||'-MFG_PART_NUM-' ||
mpn.mfg_part_num ||'-SUB_INV-' ||
msinv.secondary_inventory_name ||'-
ASSET_BOM-' ||NVL(asset_bom, 'NO_VAL'))
```

Data Set Key: `eam-store-wo`

Data Set Name

Stores Work Order (Maintenance)

Description (Purpose of the Data Set)

Contains work order details at the material level, such as allocated, issued, shortage and so on.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_requirement_operations
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id
wro.operation_seq_num
wro.inventory_item_id
```

ECC_SPEC_ID Concatenation Structure

```
to_char (
 wdj.wip_entity_id || '-' ||
 wro.operation_seq_num || '-' ||
 wro.inventory_item_id)
```

Data Set Key: `eam-asset`

Data Set Name

Assets (Maintenance)

Description (Purpose of the Data Set)

Asset information along with work orders containing those assets.

Oracle E-Business Suite Table Names

```
csi_item_instances,
wip_discrete_jobs,
wip_eam_period_balances
```

ECC_SPEC_ID Columns

```
cii.instance_id,
wdj.wip_entity_id,
period_name
```

ECC_SPEC_ID Concatenation Structure

```
((wdj.wip_entity_id || '-' ||
workorder_cost.period_name) or To_char
(cii.instance_id)
```

Data Set Key: `eam-asset-attrs`

Data Set Name

Asset Attributes (Maintenance)

Description (Purpose of the Data Set)

Asset attribute descriptive flexfield (DFF) information.

Oracle E-Business Suite Table Names

```
MTL_EAM_ASSET_ATTR_VALUES
```

ECC_SPEC_ID Columns

```
meaa.maintenance_object_id
meaa.inventory_item_id
meaa.organization_id
meaa.association_id
```

ECC_SPEC_ID Concatenation Structure

```
meaa.maintenance_object_id ||' || '''-''' || '||
meaa.inventory_item_id ||' || '''-''' || '||
meaa.organization_id ||' || '''-''' || '||
meaa.association_id
```

Data Set Key: `eam-wo-wop`

Data Set Name

Work Order Operations (Maintenance)

Description (Purpose of the Data Set)

Contains work order information at operation, resource, and instance level.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs,
wip_operations,
wip_operation_resources,
wip_op_resource_instances
```

ECC_SPEC_ID Columns

```
wdj.wip_entity_id ,
wo.operation_seq_num,
wor.resource_seq_num ,
wor.resource_id ,
wori.instance_id
```

ECC_SPEC_ID Concatenation Structure

```
( wdj.wip_entity_id
|| '-'
|| wo.operation_seq_num
|| '-'
|| wor.resource_seq_num
|| '-'
|| wor.resource_id
|| '-'
|| wori.instance_id )
```

Data Set Key: `eam-asset-details`

Data Set Name

Assets (Maintenance)

Description (Purpose of the Data Set)

Asset information

Oracle E-Business Suite Table Names

```
csi_item_instances
```

ECC_SPEC_ID Columns

```
cii.instance_id
```

ECC_SPEC_ID Concatenation Structure

```
To_char (cii.instance_id)
```

Field Service

Data Set Key: `csf-tasks`

Data Set Name

Field Service Tasks

Description (Purpose of the Data Set)

Field service tasks and its service request information.

Oracle E-Business Suite Table Names

```
jtf_tasks_b, jtf_tasks_tl, jtf_task_types_b, jtf_task_types_tl,
hz_parties, hz_party_sites, hz_locations, jtf_task_priorities_tl,
jtf_task_statuses_b, jtf_task_statuses_tl, cs_incidents_all_b,
cs_incidents_all_tl, cs_incident_types_tl, cs_incident_statuses_tl,
cs_incident_severities_tl, mtl_system_items_b_kfv, mtl_system_items_tl,
mtl_categories_kfv, csi_item_instances ,csf_access_hours_b,
jtf_task_assignments, jtf_task_audits_b, csp_requirement_headers ,
csp_requirement_lines, csf_required_skills_b
```

ECC_SPEC_ID Columns

```
task_id
```

ECC_SPEC_ID Concatenation Structure

```
task_id
```

G-Invoicing

Data Set Key: `IGT_ORDER`

Data Set Name

Projects: G-Invoicing

Description (Purpose of the Data Set)

This function facilitates view of General Terms & Conditions (GT&Cs) and associated Orders information. It enables a user to pull new orders, process them for approval, and push them back to G-Invoicing. It also helps the user execute orders through their association with Oracle Projects.

Oracle E-Business Suite Table Names

```
pa_igt_gtc_base,
pa_igt_order,
pa_igt_order_line,
pa_igt_order_line_sched
```

ECC_SPEC_ID Columns

```
gtc_number,
order_number,
order_line_number,
order_schedule_number
```

ECC_SPEC_ID Concatenation Structure

```
g.gtc_number
|| '~'
|| o.order_number
|| '~'
|| ol.order_line_number
|| '~'
|| ols.order_schedule_number
```

General Ledger

Data Set Key: `gl-aa`

Data Set Name

GL Account Analysis

Description (Purpose of the Data Set)

All GL journals for open periods.

In V10, closed periods are added based on profile value.

Oracle E-Business Suite Table Names

```
gl_je_batches gjb
gl_je_headers gjh
gl_je_lines gjl
```

ECC_SPEC_ID Columns

```
gjb.je_batch_id
gjh.je_header_id
gjl.je_line_num
```

ECC_SPEC_ID Concatenation Structure

```
gjb.je_batch_id || '-'|| gjh.je_header_id
|| '-' || gjl.je_line_num
```

Data Set Key: `gl-imp`

Data Set Name

GL Journal Import

Description (Purpose of the Data Set)

All GL interface lines for open periods.

Oracle E-Business Suite Table Names

```
gl_interface
gl_ledgers
gl_period_statuses
```

ECC_SPEC_ID Columns

```
gli.ROWID
gled.LEDGER_ID
gled.CHART_OF_ACCOUNTS_ID
gli.SET_OF_BOOKS_ID
gli.USER_JE_SOURCE_NAME
gli.USER_JE_CATEGORY_NAME
gps.PERIOD_NAME
```

ECC_SPEC_ID Concatenation Structure

```
gli.ROWID
                    || '-'
                    || gled.LEDGER_ID
                    || '-'
                    || gled.CHART_OF_ACCOUNTS_ID
                    || '-'
                    || gli.SET_OF_BOOKS_ID
                    || '-'
                    || gli.USER_JE_SOURCE_NAME
                    || '-'
                    || gli.USER_JE_CATEGORY_NAME
                    || '-'
                    || gps.PERIOD_NAME
```

Human Resources

Data Set Key: PER

Data Set Name

Human Resources

Description (Purpose of the Data Set)

This data set is used to fetch People, Assignment, Address, Phone, Qualification, Competency, TAGs, and Performance data.

Oracle E-Business Suite Table Names

```
PER_ECC_PEOPLE_V, PER_ECC_ASSIGNMENT_V, PER_ECC_ADDRESS_V,
PER_ECC_COMPETENCY_V, PER_ECC_PHONE_V,
PER_ECC_QUALIFICATION_V, PER_ECC_TAG_V,
PER_ECC_PERFORMANCE_V
```

ECC_SPEC_ID Columns

```
person_id, granted_user_id, security_profile_id
```

ECC_SPEC_ID Concatenation Structure

```
person_id || '#' || granted_user_id || '#' || security_profile_id
```

Incentive Compensation

Data Set Key: cn-cp

Data Set Name

Recent Jobs

Description (Purpose of the Data Set)

Streamlines processing by bringing information from concurrent manager and online jobs across all phases of OIC processing into a single actionable dashboard.

Oracle E-Business Suite Table Names

```
CN_PROCESS_BATCHES_ALL
CN_CALC_SUBMISSION_BATCHES_ALL
FND_CONCURRENT_REQUESTS
```

ECC_SPEC_ID Columns

```
CN_PROCESS_BATCHES_ALL.LOGICAL_BATCH_ID
FND_CONCURRENT_REQUESTS.REQUEST_ID
CN_CALC_SUBMISSION_BATCHES_ALL.CALC_SUB_BATCH_ID
```

ECC_SPEC_ID Concatenation Structure

```
CN_PROCESS_BATCHES_ALL.LOGICAL_BATCH_ID || '' ECC_SPEC_ID,
FND_CONCURRENT_REQUESTS.REQUEST_ID || '_' ECC_SPEC_ID,
CN_CALC_SUBMISSION_BATCHES_ALL.CALC_SUB_BATCH_ID ||'__' ECC_SPEC_ID,
```

Data Set Key: `cn-trx`

Data Set Name

Sales Transaction

Description (Purpose of the Data Set)

- Bring together transactional information across all phases into a single actionable dashboard.
- Assess and manage the completeness and accuracy of calculated earnings, payments and paysheets.

Oracle E-Business Suite Table Names

```
CN_COMM_LINES_API
CN_COMMISSION_HEADERS
CN_COMMISSION_LINES
CN_PAYMENT_TRANSACTION
CN_PAYMENT_WORKSHEETS
CN_PAYRUNS
```

ECC_SPEC_ID Columns

```
cn_comm_lines_api_all.COMM_LINES_API_ID
CN_COMMISSION_LINES.COMMISSION_LINE_ID
CN_PAYMENT_TRANSACTIONS_ALL.PAYMENT_TRANSACTION_ID
cn_payment_worksheets_ALL.PAYMENT_WORKSHEET_ID
```

ECC_SPEC_ID Concatenation Structure

```
cn_comm_lines_api_all.COMM_LINES_API_ID || '_API') || '_' ||
CN_COMMISSION_LINES.COMMISSION_LINE_ID || '_LINE' || '_' ||
CN_PAYMENT_TRANSACTIONS_ALL.PAYMENT_TRANSACTION_ID || '_' ||
cn_payment_worksheets_ALL.PAYMENT_WORKSHEET_ID || '_PAY'
AS ECC_SPEC_ID
```

Data Set Key: `cn-quota`

Data Set Name

Quota and Attainment

Description (Purpose of the Data Set)

- Analyze compensation attainment and performance trends across roles, plans and periods.

- Compare quotas versus actual achievement across periods by compensation plans and plan elements.

Oracle E-Business Suite Table Names

```
cn_srp_period_quotas_all
cn_srp_per_quota_rc_all
cn_srp_quota_rules_all
cn_quota_rules_all
cn_quotas_all
cn_srp_quota_assigns_all
cn_srp_plan_assigns_all
```

ECC_SPEC_ID Columns

```
cn_srp_period_quotas_all.srp_period_quota_id
cn_srp_per_quota_rc_all.srp_per_quota_rc_id
```

ECC_SPEC_ID Concatenation Structure

```
cn_srp_period_quotas_all.srp_period_quota_id || '_' ||
cn_srp_per_quota_rc_all.srp_per_quota_rc_id
```

Data Set Key: `cn-net`

Data Set Name

Compensation Plan Visualizer

Description (Purpose of the Data Set)

This dataset is used in the Compensation Plan Visualizer.

Oracle E-Business Suite Table Names

```
cn_srp_per_quota_rc_all
cn_srp_quota_rules_all
cn_quota_rules_all
cn_quotas_all
cn_srp_period_quotas_all
cn_srp_quota_assigns_all
cn_srp_plan_assigns_all
cn_comp_plans_all
cn_interval_types_all_tl
hr_all_organization_units_tl
jtf_rs_resource_extns_tl
jtf_rs_salesreps
cn_cal_per_int_types_all
cn_period_statuses_all
cn_revenue_classes_all
jtf_rs_roles_tl
cn_calc_formulas_all
cn_calc_sql_exps_all
cn_formula_inputs_all
cn_rt_quota_asgns_all
cn_rate_schedules_all
hr_operating_units
cn_salesreps
CN_SRP_PMT_PLANS_ALL
CN_PMT_PLANS
CN_SRP_PAY_GROUPS_ALL
cn_pay_groups_all
JTF_RS_RESOURCE_EXTNS_VL
```

ECC_SPEC_ID Columns

```
cn_srp_period_quotas_all.srp_period_quota_id,
cn_srp_per_quota_rc_all.srp_per_quota_rc_id
```

ECC_SPEC_ID Concatenation Structure

```
cn_srp_period_quotas_all.srp_period_quota_id
|| '_'||
cn_srp_per_quota_rc_all.srp_per_quota_rc_id
```

iProcurement

Data Set Key: `icx-catalog`

Data Set Name

iProcurement Catalog

Description (Purpose of the Data Set)

Provides information about all the items in the catalog.

Oracle E-Business Suite Table Names

```
icx_cat_items_ctx_hdrs_tlp ctx
po_system_parameters_all psp,
icx_cat_attribute_values av,
icx_cat_attribute_values_tlp avtlp,
ap_supplier_sites_all pvs,
mtl_categories_kfv mck,
mtl_system_items_kfv msikfv,
mtl_units_of_measure_tl muomtl,
gl_sets_of_books gsob,
financials_system_params_all fsp,
fnd_lookup_values flv,
icx_rvw_reviews_summary_tbl reviews,
icx_cat_categories_tl cat_tl
icx_cat_punchout_items ctx,
icx_cat_pch_item_attrs av,
icx_cat_pch_item_attrs_tlp avtlp,
ap_suppliers vendor,
icx_cat_content_zones_b zones,
icx_cat_punchout_zone_details dtls,
icx_cat_content_zones_tl tl,
por_noncat_templates_all_b b,
por_noncat_templates_all_tl tl,
po_vendors pv
```

ECC_SPEC_ID Columns

```
ctx.inventory_item_id,
ctx.po_line_id,
ctx.req_template_name,
ctx.req_template_line_num,
ctx.org_id,
ctx.punchout_item_id,
b.zone_id,
b.template_id
```

ECC_SPEC_ID Concatenation Structure

1. `ctx.inventory_item_id || '#' || ctx.po_line_id || '#' || ctx.req_template_name || '#' || ctx.req_template_line_num || '#' || ctx.org_id "ECC_SPEC_ID"`
2. `TO_CHAR(ctx.punchout_item_id) ecc_spec_id,`
3. `TO_CHAR(b.zone_id) ecc_spec_id`
4. `TO_CHAR(b.template_id) ecc_spec_id`

Data Set Key: `icx-my-notifs`

Data Set Name

My Notifications

Description (Purpose of the Data Set)

This data set provides the top five notifications for the user.

Oracle E-Business Suite Table Names

```
wf_notifications  WN
wf_item_types_tl WIT
wf_lookups_tl WL
wf_user_roles WUR
```

ECC_SPEC_ID Columns

```
WN.notification_id
```

ECC_SPEC_ID Concatenation Structure

```
WN.notification_id
```

Data Set Key: `icx-my-reqs`

Data Set Name

My Requisitions

Description (Purpose of the Data Set)

This data set provides the top five requisitions created by a user.

Oracle E-Business Suite Table Names

```
po_requisition_headers_all prh
per_all_people_f hre
hr_all_organization_units_tl org
```

ECC_SPEC_ID Columns

```
prh.requisition_header_id
```

ECC_SPEC_ID Concatenation Structure

```
prh.requisition_header_id
```

Data Set Key: `icx-top-categories`

Data Set Name

iProcurement Top Categories

Description (Purpose of the Data Set)

This data set provides information about top categories of items ordered by the requestors.

Oracle E-Business Suite Table Names

```
 po_requisition_lines_all prl
ICX_POR_CATEGORY_DATA_SOURCES cat_map
icx_cat_categories_tl cat_tl
 per_all_people_f paf
```

ECC_SPEC_ID Columns

```
cat_tl.rt_category_id
```

ECC_SPEC_ID Concatenation Structure

```
cat_tl.rt_category_id
```

Data Set Key: `icx-freq-items`

Data Set Name

iProcurement Frequently Requested Items

Description (Purpose of the Data Set)

This data set provides information about the frequently requested items in the catalog.

Oracle E-Business Suite Table Names

```
po_requisition_lines_all porl
po_lines_all pol
per_all_people_f paf
```

ECC_SPEC_ID Columns

```
porl.item_id
pol.po_line_id
porl.req_template_name
porl.req_template_line_num
porl.org_id
porl.punchout_item_id
```

ECC_SPEC_ID Concatenation Structure

1. ```
   porl.item_id || '#' || pol.po_line_id || '#'|| porl.
   req_template_name || '#'|| por.req_template_line_num || '#' || porl.
   org_id
   ```

2. ```
   TO_CHAR(porl.punchout_item_id)
   ```

Data Set Key: `icx-news`

Data Set Name

iProcurement Purchasing News

Description (Purpose of the Data Set)

This data set provide details about Purchasing News.

Oracle E-Business Suite Table Names

```
fnd_lookup_values
```

ECC_SPEC_ID Columns

```
lookup_type
lookup_code
```

ECC_SPEC_ID Concatenation Structure

```
lookup_type||'#'||lookup_code ECC_SPEC_ID
```

iStore

Data Set Key: `ibe-search`

Data Set Name

iStore Search

Description (Purpose of the Data Set)

This data set provides information about all the items of the minisite.

Oracle E-Business Suite Table Names

```
IBE_SECTION_SEARCH_PART,
MTL_SYSTEM_ITEMS_B,
IBE_MSITES_TL,
IBE_DSP_SECTIONS_TL,
MTL_SYSTEM_ITEMS_TL,
IBE_ECC_SEARCH_PART_PRICE,
MTL_CROSS_REFERENCES_V,
MTL_CUSTOMER_ITEM_XREFS_V,
JTF_AMV_ITEMS_VL,
IBE_DSP_LGL_PHYS_MAP,
IBE_DSP_FEATURED_MEDIA
```

ECC_SPEC_ID Columns

```
MINISITE_ID,SECTION_ID, ORGANIZATION_ID, INVENTORY_ITEM_ID,
PRICE_LIST_ID, CUST_ACCT_ID, PARTY_ID
```

ECC_SPEC_ID Concatenation Structure

```
MINISITE_ID || '-' ||SECTION_ID || '-' || ORGANIZATION_ID || '-' ||
INVENTORY_ITEM_ID ||'-' ||PRICE_LIST_ID || '-' || CUST_ACCT_ID || '-' ||
PARTY_ID
```

Landed Cost Management

Data Set Key: `LCM_SHIPMENTS`

Data Set Name

LCM Shipments

Description (Purpose of the Data Set)

This data set provides details for LCM Shipments and its details.

Oracle E-Business Suite Table Names

```
inl_ship_headers_all
inl_ship_lines_all
inl_tax_lines
inl_charge_lines
inl_allocations
inl_matches
fnd_lookup_values
fnd_lookup_values
fnd_lookup_values
inl_ship_types_tl
inl_ship_types_b
inl_ship_line_groups
gl_ledgers
rcv_shipment_headers
rcv_shipment_lines
rcv_transactions
mtl_system_items_tl
mtl_system_items_kfv
inl_ship_lines_all
mtl_parameters
hr_all_organization_units
hr_locations_all
hz_parties
hz_party_sites
hz_locations
po_line_locations_all
po_lines_all
po_headers_all
per_all_people_f
mtl_categories_tl
mtl_categories_kfv
hr_operating_units
inl_ship_line_types_tl
pon_price_element_types_tl
```

ECC_SPEC_ID Columns

```
ish.ship_header_id
islg.ship_line_group_id
isl.parent_ship_line_id
isl.ship_line_id
aa.component_type_code
aa.component_reference
```

ECC_SPEC_ID Concatenation Structure

```
ish.ship_header_id
|| '-'
|| islg.ship_line_group_id
|| '-'
|| nvl (isl.parent_ship_line_id,isl.ship_line_id)
|| '-'
|| aa.component_type_code
|| '-'
|| aa.component_reference
```

Lease and Finance Management

Data Set Key: `okl-ast-ctr-dtls okl-okl`

Data Set Name

Contracts

Description (Purpose of the Data Set)

Lease Asset Contract Details

Oracle E-Business Suite Table Names

```
okc_k_headers_all_b chr, okc_k_headers_tl chrt, okl_k_headers khr,
hr_operating_units hr_ou, okc_k_party_roles_b cust, hz_parties hp,
hz_cust_accounts hca, okc_statuses_b sts, okc_statuses_tl ststl,
okl_products product, okc_rule_groups_b rg_latown, okc_rules_b r_latown,
okl_product_parameters_all_v ppv, okc_rule_groups_b rg_laevel,
okc_rules_b r_laevel, okc_rule_groups_b rg_amteoc, okc_rules_b r_amteoc,
okc_rule_groups_b rg_amtfoc, okc_rules_b r_amtfoc_ambpoc, okc_rules_b
r_amtfoc_amtinv, okl_la_bill_to_uv hcsu,po_vendors pv, okl_k_headers
khr, okc_k_party_roles_b cpl_ml , okc_k_items, okc_k_lines_b cle_ml ,
okc_statuses_b sts, okc_statuses_tl sts_tl, mtl_system_items_tl msit,
hz_cust_site_uses_all csu, fnd_lookup_values lk_asset_type,
okl_asset_returns_all_b asset_rma, mtl_system_items_b rmk_item_hdr,
okl_asset_cndtns_all ast_cond
```

ECC_SPEC_ID Columns

```
hdr.id,asset_line_id
```

ECC_SPEC_ID Concatenation Structure

```
hdr.id||'-'||ln.asset_line_id ecc_spec_id
```

Data Set Key: `okl-ast-qte-dts`

Data Set Name

Quotes

Description (Purpose of the Data Set)

Lease Asset Quote Details

Oracle E-Business Suite Table Names

```
okl_trx_quotes_all_b qte_hdr, okl_trx_quotes_tl qte_hdr_tl,
okl_lease_quotes_b lse_qte, okl_lease_quotes_tl lse_qte_tl,
fnd_currencies_tl fnd_curr, okc_k_headers_all_b khr_hdr, okl_k_headers
okl_khr_hdr, okc_k_headers_all_b target_khr_hdr, okl_products prod,
okc_statuses_tl khr_sts_tl, hz_cust_accounts_all qte_cust_hdr,
hz_parties qte_party_hdr, fnd_lookup_values qte_sts_lkup,
fnd_lookup_values qte_typ_lkup, fnd_lookup_values qte_rsn_lkup,
okl_assets_b lse_asset, okl_asset_components_b lse_assetcomp,
okc_k_lines_b kle, okl_k_lines cle, okc_k_lines_tl kle_tl,
okc_statuses_tl kle_sts_tl
```

ECC_SPEC_ID Columns

```
Quote_id, Quote_line_id
```

ECC_SPEC_ID Concatenation Structure

```
to_char(QTD.Quote_id)||'-'||to_char(QTD.Quote_line_id)
```

Data Set Key: `okl-ar-trx`

Data Set Name

Lease Outstanding Receivables

Description (Purpose of the Data Set)

All the Lease Receivables invoices and credit memos.

Oracle E-Business Suite Table Names

```
ar_payment_schedules_all ps
ra_salesreps_all
jtf_rs_resource_extns_vl (View)
ra_cust_trx_types_all
ra_batch_sources_all
ra_customer_trx_all trx
hz_cust_accounts
hz_parties
hz_cust_site_uses_all
hr_all_organization_units_tl
fnd_lookup_values
gl_ledgers
fnd_currencies_tl
gl_period_statuses
ra_customer_trx_lines_all a
ar_receipt_methods
```

ECC_SPEC_ID Columns

```
ps.payment_schedule_id,
trx.customer_trx_id,
a.customer_trx_line_id
```

ECC_SPEC_ID Concatenation Structure

```
 'TRX-' || ps.payment_schedule_id
|| '-' || trx.customer_trx_id || '-' || a.customer_trx_line_id
```

Data Set Key: `okl-ar-trx-lines`

Data Set Name

Lease Invoice Lines

Description (Purpose of the Data Set)

Lease Invoices Lines

Oracle E-Business Suite Table Names

```
OKL_BPD_AR_INV_LN_DTLS_V a (view)
ra_customer_trx_lines_all trxl
ra_customer_trx_all trx
ar_payment_schedules_all ps
hr_all_organization_units_tl org
```

ECC_SPEC_ID Columns

```
a.customer_trx_id,
a.customer_trx_line_id
```

ECC_SPEC_ID Concatenation Structure

```
a.customer_trx_id || '-'
a.customer_trx_line_id
```

Data Set Key: `okl-ar-trx-appl`

Data Set Name

Lease Invoice Applications

Description (Purpose of the Data Set)

Lease Invoice Applications

Oracle E-Business Suite Table Names

```
okl_receipt_applications_uv appl, (View)
ar_cash_receipts_all arcash,
hr_all_organization_units_tl org,
fnd_currencies_tl trans_curr_tl
```

ECC_SPEC_ID Columns

```
appl.cash_receipt_id,
appl.receivable_application_id,
customer_trx_line_id
```

ECC_SPEC_ID Concatenation Structure

```
appl.cash_receipt_id || '-' ||
appl.receivable_application_id || '-' ||
customer_trx_line_id
```

Data Set Key: `okl-ar-cm-appl`

Data Set Name

Credit Memo Applications

Description (Purpose of the Data Set)

Lease Credit Memo Applications

Oracle E-Business Suite Table Names

```
ar_receivable_applications_all a,
ar_payment_schedules_all b,
ra_customer_trx_all c
```

ECC_SPEC_ID Columns

```
c.customer_trx_id,
a.RECEIVABLE_APPLICATION_ID
```

ECC_SPEC_ID Concatenation Structure

```
c.customer_trx_id || '-' || a.RECEIVABLE_APPLICATION_ID
```

Data Set Key: `okl-cash`

Data Set Name

Lease Receipt

Description (Purpose of the Data Set)

Lease Receipt Details

Oracle E-Business Suite Table Names

```
ar_cash_receipts_all         receipt,
ar_payment_schedules_all     ps,
hr_all_organization_units_tl org,
hz_cust_accounts             account,
hz_parties                   party,
gl_ledgers                   ledger,
ar_cash_receipt_history_all  history_first,
ar_receipt_methods           rcpt_methods,
hz_cust_site_uses_all        u_bill,
FND_LOOKUP_VALUES            lookup_status,
FND_LOOKUP_VALUES            lookup_type,
FND_CURRENCIES_TL            receipt_curr_tl,
GL_PERIOD_STATUSES           glps,
OKL_TRX_CSH_Rcpt_all_B oklrcpt,
OKL_TXL_RCPT_APPS_ALL_B oklallc,
okc_k_headers_all_b okch
```

ECC_SPEC_ID Columns

```
receipt.cash_receipt_id,
oklrcpt.ID
```

ECC_SPEC_ID Concatenation Structure

```
'ARCASHRCPT-' || receipt.cash_receipt_id || '-' || oklrcpt.ID
```

Data Set Key: `okl-cash-appl`

Data Set Name

Lease Receipt Applications

Description (Purpose of the Data Set)

Lease Receipt Application Details

Oracle E-Business Suite Table Names

```
okl_receipt_applications_uv appl, (View)
ar_cash_receipts_all arcash,
hr_all_organization_units_tl org,
fnd_currencies_tl trans_curr_tl
```

ECC_SPEC_ID Columns

```
appl.cash_receipt_id,
appl.receivable_application_id,
customer_trx_line_id
```

ECC_SPEC_ID Concatenation Structure

```
appl.cash_receipt_id || '-' ||
appl.receivable_application_id || '-' ||
customer_trx_line_id
```

Data Set Key: `okl-cash-alloc`

Data Set Name

Lease Receipt Allocations

Description (Purpose of the Data Set)

Lease Receipt Allocation Details

Oracle E-Business Suite Table Names

```
ar_cash_receipts_all arcash,
hr_all_organization_units_tl org,
fnd_currencies_tl trans_curr_tl,
OKL_TRX_CSH_Rcpt_all_B trx,
OKL_TXL_RCPT_APPS_ALL_B trxl,
okl_strm_type_b sty,
okc_k_headers_all_b chr
```

ECC_SPEC_ID Columns

```
arcash.cash_receipt_id,
trxl.id
```

ECC_SPEC_ID Concatenation Structure

```
arcash.cash_receipt_id || '-' || trxl.id
```

Data Set Key: `okl-unapp-cash`

Data Set Name

Lease Unapplied Cash

Description (Purpose of the Data Set)

Lease Unapplied Cash Details

Oracle E-Business Suite Table Names

```
 ar_cash_receipts_all         receipt,
ar_payment_schedules_all     ps,
hr_all_organization_units_tl org,
hz_cust_accounts             account,
hz_parties                   party,
gl_ledgers                   ledger,
ar_cash_receipt_history_all  history_first,
ar_receipt_methods           rcpt_methods,
hz_cust_site_uses_all        u_bill,
FND_LOOKUP_VALUES            lookup_status,
FND_LOOKUP_VALUES            lookup_type,
FND_CURRENCIES_TL            receipt_curr_tl,
GL_PERIOD_STATUSES           glps,
OKL_TRX_CSH_Rcpt_all_B oklrcpt,
OKL_TXL_RCPT_APPS_ALL_B oklallc,
okc_k_headers_all_b okch
```

ECC_SPEC_ID Columns

```
receipt.cash_receipt_id,
oklrcpt.ID
```

ECC_SPEC_ID Concatenation Structure

```
'ARCASHRCPT-' || receipt.cash_receipt_id || '-' || oklrcpt.ID
```

Data Set Key: `okl-unapp-cash_inv`

Data Set Name

Lease Unapplied Cash Invoices

Description (Purpose of the Data Set)

Lease Open Invoices

Oracle E-Business Suite Table Names

```
 OKL_BPD_AR_INV_LN_DTLS_V a, (View)
ra_customer_trx_all b,
okc_k_lines_tl c,
ar_payment_schedules_all d,
hz_parties p,
hz_cust_accounts hzca,
gl_ledgers ledger
```

ECC_SPEC_ID Columns

```
a.CUSTOMER_TRX_ID,
a.CUSTOMER_TRX_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
a.CUSTOMER_TRX_ID || '-' || a.CUSTOMER_TRX_LINE_ID
```

Data Set Key: `okl_arcon_dtls`

Data Set Name

Contract Details

Description (Purpose of the Data Set)

Lease Contract Details

Oracle E-Business Suite Table Names

```
okc_k_headers_all_b chr,
       okl_trx_contracts_all trx,
       okl_txl_cntrct_lns_all txl,
       okl_trns_acc_dstrs_all dst,
       okl_trx_types_b try,
       okl_trx_types_tl try_tl,
       okl_strm_type_b str,
       xla_distribution_links a,
       xla_ae_lines c,
       gl_code_combinations gcc,
UNION ALL
       hz_cust_accounts_all    customer,
       hz_parties              party,
       gl_ledgers             gll,
       okl_k_headers hdr,
       ra_customer_trx_all trx,
        ra_customer_trx_lines_all txl,
        RA_CUST_TRX_LINE_GL_DIST_ALL dist,
        hz_cust_accounts_all    customer,
        hz_parties              party,
        okl_strm_type_b         sty,
        okl_sys_acct_opts_all  osa,
        gl_ledgers             gll,
        OKL_TRX_AR_INVOICES_B kbh,
        OKL_TXL_AR_INV_LNS_B kbl,
        OKL_TXD_AR_LN_DTLS_B kbd,
        okl_trx_types_b typ,
        okl_trx_types_tl try,
UNION ALL
okl_streams strm,
okl_strm_elements se,
OKL_PROD_STRM_TYPES acc_str,
okl_strm_type_b str,
okl_report_trx_params  ort,
okl_reports_b          orb,
fnd_flex_values     ffv,
okl_sys_acct_opts_all  osa,
xla_transaction_entities xte,
UNION ALL
 xla_events xe,
xla_ae_headers aeh,
xla_ae_lines ael,
okl_trx_ap_invs_all_b aph,
okl_txl_ap_inv_lns_all_b apl,
ap_invoices_all inv,
ap_invoice_lines_all invl
OKL_CNTRCT_BALANCES_ALL bal
```

ECC_SPEC_ID Columns

```
chr.id, xla.event_id,gcc.code_combination_id,str.idhdr.id, dist,
cust_trx_line_gl_dist_id, dist.code_comination_id,

hdr.id, dist.cust_trx_line_gl_dist_id,dist_code_combination_id

chr.id,sty.code

apl.kle_id, try_tl.name,sty.code,xe.event_id, gcc.code_combination_id,
invl.line_number,aph.fund_req_num
```

ECC_SPEC_ID Concatenation Structure

```
chr.contract_id||'-'||xla.event_id||'-'||gcc.
code_combination_id||'-'||str.id
UNION ALL
hdr.contract_id||'-'||dist.CUST_TRX_LINE_GL_DIST_ID||'-'||dist.
code_combination_id
UNION ALL
rec.contract_id||'-'||'0'||'-'||'0'||'-'||sty.code||'-'||any one of
(ACC_STM_BAL,BILL_STM_BAL,RESIDUAL,LOAN_PRIN_BAL
UNION ALL
apl.kle_id||'-'||.trx_name||'-'||sty.stream_code;  --||'-'||nvl(xe..
event_id,0);
'||nvl(gcc.code_combination_id,0)||'-'||nvl(invl.line_number,0);
||'-'||aph.fund_req_num||'-'||0;
```

Data Set Key: `okl_arcon_quote_summ`

Data Set Name

Lease Quote Summary

Description (Purpose of the Data Set)

Lease Quote Summary

Oracle E-Business Suite Table Names

```
okl_ecc_arcon_quotes
```

ECC_SPEC_ID Columns

```
contract_id,stream_code,asset_number,
gl_account
```

ECC_SPEC_ID Concatenation Structure

```
contract_id||'-'||stream_code||'-'||asset_number||'-'||gl_account
```

Data Set Key: `okl_arcon_quote_dtls`

Data Set Name

Lease Contract Quote Details

Description (Purpose of the Data Set)

Lease Contract Quote Details

Oracle E-Business Suite Table Names

```
okl_trx_quotes_all_b qth,
       okc_k_headers_all_b okc,
       OKC_K_LINES_B CLEB,
       OKC_K_LINES_TL CLET,
       OKL_K_LINES KLE,
       OKL_TXL_qte_lines_all_b QLB,
       OKL_TXL_QUOTE_LINES_TL qtl,
       okl_formulae_b c,
       OKL_TRX_TYPES_B f,
       OKL_TRX_TYPES_TL TRYT,
       okl_sys_acct_opts_all  osa,
       gl_ledgers             gll,
       okl_reports_b          orb,
       okl_report_trx_params  ort,
       fnd_flex_values     ffv,
       okl_strm_type_b sty,
       okl_k_headers khr,
       okl_product_parameters_all_v ppv
```

ECC_SPEC_ID Columns

```
qth.khr_id,qth.quote_number,
qlb.line_number,c.name,
```

ECC_SPEC_ID Concatenation Structure

```
qth.contract_id||'-'||qth.quote_number||'-'||qlb.line_number||'-'||c.
name,
```

Data Set Key: `okl_arcon_cont_summ`

Data Set Name

Contract Account Balances Summary

Description (Purpose of the Data Set)

Contract Account Balances Summary

Oracle E-Business Suite Table Names

```
okl_ecc_acct_recon ecc,
okl_report_trx_params ort,
okl_reports_b orb,
okl_strm_type_b sty,
fnd_flex_values ffv
okl_sys_acct_opts_all  osa,
gl_ledgers             gll,
okl_k_headers           khr,
okl_product_parameters_all_v ppv,
okl_k_rate_params orp
```

ECC_SPEC_ID Columns

```
ecc.contract_number,ecc.gl_account,ort.account_group_code,ecc.
stream_code
```

ECC_SPEC_ID Concatenation Structure

```
ecc.contract_number||'-'||ecc.gl_account||'-'||ort.
account_group_code||'-'||ecc.stream_code
```

Data Set Key: `okl_arcon_contract_revenue`

Data Set Name

Contract Exception Details

Description (Purpose of the Data Set)

Contract Exception Details

Oracle E-Business Suite Table Names

```
okl_streams strm,
okl_strm_elements se,
OKL_PROD_STRM_TYPES acc_str, okl_k_headers khdr,
okl_strm_type_b sty,
ra_customer_trx_lines_all  a,
okl_strm_type_b b,
okc_k_headers_all_b hdr,
OKL_PROD_STRM_TYPES acc_str, okl_k_headers khdr,
okl_strm_type_b sty,
okl_strm_type_b sty1,
okl_st_gen_tmpt_lns_all gtlv,
okl_st_gen_templates_all gttv,
okl_st_gen_tmpt_sets_all gtsv,
okl_ae_tmpt_sets_all aes,
okl_products_v pdt,
okl_strm_type_b sty,
okl_ext_pay_invs_all_b ep,
okl_trx_ap_invs_all_b tap,
okl_txl_ap_inv_lns_all_b lns
okl_ext_pay_invs_all_b ep,
okl_trx_ap_invs_all_b tap,
okl_txl_ap_inv_lns_all_b lns
okl_trx_contracts_all,
ra_interface_errors_all a,
ra_interface_lines_all b,
okl_trx_ap_invs_all_b,
okl_trx_ap_invs_all_b
xla_transaction_entities xte,
xla_events xe,
okl_trx_ar_invoices_b trxar,
ra_customer_trx_all trx, ra_customer_trx_lines_all txl,
xla_transaction_entities xte,
xla_events xe,
ap_interface_rejections a,
ap_invoices_interface b,
ap_invoice_lines_interface c,
okl_txl_ap_inv_lns_all_b d,
xla_transaction_entities xte,
xla_events xe,
OKL_TXL_AP_INV_LNS_ALL_B apl, ap_invoices_all inv,
ap_invoice_lines_all invl,
okc_k_headers_all_b c,
okl_sys_acct_opts_all  osa,
gl_ledgers             gll,
okl_product_parameters_all_v ppv
```

ECC_SPEC_ID Columns

```
c.contract_number
```

ECC_SPEC_ID Concatenation Structure

```
c.contract_number
```

Data Set Key: `okl-vendor-cash`

Data Set Name

Lease Vendor Cash Flow

Description (Purpose of the Data Set)

Lease Vendor Cash Flow Details

Oracle E-Business Suite Table Names

```
okc_k_lines_b           cle,
okc_k_lines_b           clet,
okc_k_headers_all_b     chr,
hz_cust_accounts_all    customer,
hz_parties              party,
okl_streams              stm,
okl_strm_elements        ste,
okl_strm_type_b          stb,
okl_party_payment_hdr    pph,
okc_k_headers_all_b      chr,
okl_party_payment_dtls   ppd
ap_supplier_sites_all    pvs,
hz_party_sites           hps,
hz_locations             hl
okl_txd_ar_ln_dtls_b             tld,
okl_trx_ar_invoices_b            tai,
okl_txl_ar_inv_lns_b             til,
ra_customer_trx_lines_all        cust_lines,
ra_customer_trx_all              cust,
ar_cash_receipts_all             acr,
ar_payment_schedules_all         sch,
ar_receivable_applications_all   app,
ar_distributions_all             ard,
ar_payment_schedules_all         aging,
ra_cust_trx_types_all            trx_type,
okl_trx_ap_invs_all_b      tap,
okl_txl_ap_inv_lns_all_b   tpl,
ap_invoices_all            api,
ap_invoice_lines_all       apil,
ap_suppliers               pov,
ap_invoice_payments_all    app,
okl_txd_ar_ln_dtls_b       tld,
ap_checks_all              apc,
ap_payment_schedules_all   aps,
ap_holds,
okl_k_party_roles       okpr,
okl_la_kle_vendors_uv   olkv
```

ECC_SPEC_ID Columns

```
chr.contract_id, cle.cle_id, cle.lse_id, ppd.vendor_id, cust_lines.
CUSTOMER_TRX_Line_Id, acr.Cash_receipt_id, app.
receivable_application_id, api.Invoice_id, apc.check_id, aps.
invoice_payment_id, apil.ap_inv_line_number
```

ECC_SPEC_ID Concatenation Structure

```
chr.contract_id||cle.cle_id||cle.lse_id||ppd.vendor_id||cust_lines.
CUSTOMER_TRX_Line_Id||acr.Cash_receipt_id||app.
receivable_application_id||api.Invoice_id||apc.check_id||aps.
invoice_payment_id||apil.ap_inv_line_number
```

Data Set Key: `okl-vcf-summary`

Data Set Name

Lease Vendor Cash Summary

Description (Purpose of the Data Set)

This data set holds summarized data for the Vendor Cash Flow Dashboard.

Oracle E-Business Suite Table Names

```
okl_ecc_vcf_summary_v
```

ECC_SPEC_ID Columns

```
contract_id,contract_number,cle_id,customer_trx_line_id,line_type
```

ECC_SPEC_ID Concatenation Structure

```
contract_id||contract_number||cle_id||customer_trx_line_id||line_type
```

Data Set Key: `okl-ecc-reads`

Data Set Name

Lease Meter Readings Information

Description (Purpose of the Data Set)

Lease Meter Readings Information

Oracle E-Business Suite Table Names

```
hr_all_organization_units_tl,
hr_organization_information,
okc_k_lines_tl,
okc_statuses_tl,
fa_categories_b cat,
fa_book_controls fbc,
fa_books fb,
fa_additions_b fa,
fa_methods fm,
okc_k_lines_b cle,
okc_k_items item,
okl_ecc_meter_sch_v esm,
csi_counters_tl cct,
qp_list_headers_tl qlht,
okl_ecc_usage_vendor_v lvp,
mtl_system_items_tl msi,
hz_parties party,
po_vendors vendor
```

ECC_SPEC_ID Columns

```
contract_id,contract_line_id,counter_id.vendor_site,period_number,
period_id
```

ECC_SPEC_ID Concatenation Structure

```
 ecv.contract_id|| ''-''|| ecv.contract_line_id|| ''-''|| ecv.
counter_id|| ''-''|| ecv.vendor_site|| ''-''|| ecv.period_number||
''-''|| nvl (ecv.id,0)
```

Data Set Key: `okl-batch-details`

Data Set Name

Lease Batch Details

Description (Purpose of the Data Set)

Lease Batch Details

Oracle E-Business Suite Table Names

```
okl_ctr_reading_batches_all crb,
okl_counter_readings ccr,
okl_ecc_usage_vendor_v oeu,
po_vendors pov,
fnd_lookup_values flv,
fnd_lookup_values flv1,
fnd_lookup_values flv2
```

ECC_SPEC_ID Columns

```
       contract_id, batch_id
```

ECC_SPEC_ID Concatenation Structure

```
''OKL - ''|| to_char (ccr.khr_id)|| to_char (crb.id)
```

Data Set Key: `okl-mtr-errors`

Data Set Name

Lease Meter Errors

Description (Purpose of the Data Set)

Lease Meter Errors

Oracle E-Business Suite Table Names

```
okl_ctr_reading_batches_all,
fa_categories_b,
fa_book_controls,
fa_books,
fa_additions_b,
fa_methods,
okc_k_lines_b,
okc_k_items,
okl_ctr_reading_errors a,
okl_counter_readings b,
okl_ecc_usage_vendor_v oeu,
csi_counters_tl cct,
mtl_system_items_tl mit
```

ECC_SPEC_ID Columns

```
error_id,meter_read_id, error sequence
```

ECC_SPEC_ID Concatenation Structure

```
to_char (a.id) || to_char (a.ocr_id) || to_char (a.error_seq_number)
```

Data Set Key: `okl-ar-couter`

Data Set Name

Lease Invoice Information

Description (Purpose of the Data Set)

Lease Invoice Information

Oracle E-Business Suite Table Names

```
 okc_k_lines_b cle, okc_k_lines_b clet, okc_k_lines_b lp,
okc_k_headers_all_b chr, hz_cust_accounts_all customer, hz_parties
party, okl_party_payment_hdr pph, okl_party_payment_dtls ppd,
ap_suppliers vendor, okc_k_headers_all_b chr, ap_supplier_sites_all pvs,
hz_party_sites hps, hz_locations hl, okl_txd_ar_ln_dtls_b tld,
okl_trx_ar_invoices_b tai, okl_txl_ar_inv_lns_b til,
ra_customer_trx_lines_all cust_lines, ra_customer_trx_all cust,
ar_cash_receipts_all acr, ar_payment_schedules_all sch,
ar_receivable_applications_all app, ar_distributions_all ard
```

ECC_SPEC_ID Columns

```
ar_inv_number,.interface_header_attribute7,
cle.lse_id,.customer_trx_id,
customer_trx_line_id,
cash_receipt_id,receivable_application_id
```

ECC_SPEC_ID Concatenation Structure

```
to_char (ar_inv_lns.ar_inv_number)|| '-'|| decode (cle.lse_id,49
,'Service',57 ,'Usage',cle.lse_id) || '-'|| ar_inv_lns.
interface_header_attribute7|| '-'|| ar_inv_lns.customer_trx_id|| '-' ||
ar_inv_lns.customer_trx_line_id|| '-'||nvl(ar_inv_lns.cash_receipt_id,0)
|| '-'||nvl(ar_inv_lns.receivable_application_id,0) || '-'||nvl(ap_atts.
invoice_num,0)
```

Logistics (Inventory)

Data Set Key: `Delivery_Details`

Data Set Name

Shipping Delivery Details

Description (Purpose of the Data Set)

This data set holds information about shipment details created by Order Management Sales orders.

Oracle E-Business Suite Table Names

```
wsh_delivery_details
```

ECC_SPEC_ID Columns

```
delivery_detail_id
```

ECC_SPEC_ID Concatenation Structure

```
delivery_detail_id
```

Data Set Key: `inv-onhand`

Data Set Name

Inventory On Hand

Description (Purpose of the Data Set)

This data set provides the warehouse and materials management teams with tools to better manage on-hand inventory and maintain efficient operations.

Oracle E-Business Suite Table Names

```
mtl_system_items_b_kfv (msib)
mtl_onhand_quantities_detail (moqd)
```

ECC_SPEC_ID Columns

```
msib.organization_id
msib.inventory_item_id
moqd.revision
moqd.subinventory_code
moqd.locator_id
moqd.lot_numbe
```

ECC_SPEC_ID Concatenation Structure

```
rmsib.organization_id || '--' || msib.inventory_item_id || '--' ||moqd.
revision || '--' || moqd.subinventory_code || '--' || moqd.locator_id ||
'--' || moqd.lot_number
```

Data Set Key: `inv-cyclecounting`

Data Set Name

Inventory Cycle Counting

Description (Purpose of the Data Set)

This data set simplifies the management of cycling counting programs.

Oracle E-Business Suite Table Names

```
mtl_cycle_count_entries (mcce)
mtl_cycle_count_headers (mcch)
mtl_parameters (mp)
mtl_cycle_count_items (mcci)
```

ECC_SPEC_ID Columns

```
mcce.cycle_count_entry_id
mcch.cycle_count_header_id
mp.organization_id
mcci.inventory_item_id
```

ECC_SPEC_ID Concatenation Structure

```
DECODE(mcce.cycle_count_entry_id, NULL, mcch.cycle_count_header_id ||
'--' || mp.organization_id || '--' || mcce.inventory_item_id, mcce.
cycle_count_entry_id)
```

Data Set Key: `inv-reservations`

Data Set Name

Reservations

Description (Purpose of the Data Set)

This data set show existing demand and, when a reservation exists, the supply associated with the demand.

Oracle E-Business Suite Table Names

```
mtl_reservations
mtl_sales_orders_kfv
oe_order_lines_all
oe_order_headers_all
gme_batch_header
gme_material_details
wip_requirement_operations
wip_entities
```

ECC_SPEC_ID Columns

```
line_id,reservation_id
batch_no,gmd.formulaline_id,mr.reservation_id
wro.wip_entity_id,wro.inventory_item_id,wro.operation_seq_num,mr.
reservation_id
```

ECC_SPEC_ID Concatenation Structure

```
line_id
|| '--'
|| reservation_id ecc_spec_id

batch_no
|| '--'
|| gmd.formulaline_id
|| '--'
|| mr.reservation_id ecc_spec_id

(wro.wip_entity_id)
|| '--'
|| (wro.inventory_item_id)
|| '--'
|| (wro.operation_seq_num)
|| '--'
|| mr.reservation_id ecc_spec_id

mr.reservation_id
```

Data Set Key: `inv-activity`

Data Set Name

Activities

Description (Purpose of the Data Set)

This data set contains the activities of resources that are executing inventory transactions that have a calculated duration.

Oracle E-Business Suite Table Names

```
wms_els_trx_src
wms_activity_details
fnd_lookup_values
mtl_parameters
mtl_system_items_kfv
fnd_user
per_all_people_f
mtl_item_locations_kfv
mtl_categories_b_kfv
wms_zones_tl
```

ECC_SPEC_ID Columns

```
wms_els_trx_src.els_trx_src_id, wms_activity_details.els_trx_src_id
```

ECC_SPEC_ID Concatenation Structure

```
wms_els_trx_src.els_trx_src_id, wms_activity_details.els_trx_src_id
```

Data Set Key: `Inbound_Details`

Data Set Name

Inbound Details

Description (Purpose of the Data Set)

Main data set for Receiving Dashboard. It contains pending receipt/pending inspection/pending putaway/dock to stock transactions data.

Oracle E-Business Suite Table Names

```
PO_HEADERS_ALL
PO_LINES_ALL
PO_LINE_LOCATIONS_ALL
PO_RELEASES_ALL
PO_VENDORS
MTL_SYSTEM_ITEMS_B_KFV
MTL_PARAMETERS
RCV_PARAMETERS
RCV_SHIPMENT_HEADERS
RCV_SHIPMENT_LINES
MTL_SUPPLY
RCV_SUPPLY
MTL_MATERIAL_TRANSACTIONS
PO_REQUISITION_HEADERS_ALL
PO_REQUISITION_LINES_ALL
OE_ORDER_HEADERS_ALL
OE_ORDER_LINES_ALL
HZ_CUST_ACCOUNTS
HZ_PARTIES
MTL_SYSTEM_ITEMS
MTL_UNITS_OF_MEASURE_TL
RCV_TRANSACTIONS
FINANCIALS_SYSTEM_PARAMS_ALL
WMS_LICENSE_PLATE_NUMBERS
WSH_DELIVERY_DETAILS
```

ECC_SPEC_ID Columns

```
PO_LINE_LOCATIONS_ALL.PO_LINE_LOCATION_ID,
RCV_SHIPMENT_LINES.SHIPMENT_LINE_ID,
OE_ORDER_LINES_ALL.LINE_ID,
RCV_SUPPLY.RCV_TRANSACTION_ID,
RCV_TRANSACTIONS.TRANSACTION_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PO_'||PLL.PO_LINE_LOCATION_ID,
RT_' || RT.transaction_id,
'REQ_' || rsl.shipment_line_id,
'INVENTORY_' || rsl.shipment_line_id,
decode(rsh.asn_type, 'LCM', 'LCM', 'ASN', 'ASN', 'ASBN', 'ASN')|| '_'||
rsl.shipment_line_id,
'RMA_' ||  OEL.LINE_ID ECC_SPEC_ID,
'RS_'||RS.RCV_TRANSACTION_ID
```

Data Set Key: `inv-trace-suppliers`

Data Set Name

Inventory Trace Suppliers

Description (Purpose of the Data Set)

This dataset contains Supply information.

Oracle E-Business Suite Table Names

```
po_headers_all,mtl_material_transactions, mtl_transaction_lot_numbers ,
po_lines_all
```

ECC_SPEC_ID Columns

```
po_headers_id,po_lines_id,transaction_id,transaction_source_id,segment1
```

ECC_SPEC_ID Concatenation Structure

```
'PO_'||to_char (pla.po_line_id
 || '-' || pt.dataset_pk_id)
```

Data Set Key: `inv-trace-inbound-onhand`

Data Set Name

Inventory Trace Inbound and On-Hand

Description (Purpose of the Data Set)

This dataset contains the item,lot and its onhand information.

Oracle E-Business Suite Table Names

```
mtl_onhand_quantities_detail,mtl_lot_numbers,mtl_system_items_b
```

ECC_SPEC_ID Columns

```
inventory_item_id,organization_id,transaction_id,
lot_number
```

ECC_SPEC_ID Concatenation Structure

```
 msit.inventory_item_id
 || '-' || msit.organization_id
 || '-' || mmt.transaction_id
```

Data Set Key: `inv-trace-move`

Data Set Name

Inventory Trace Move

Description (Purpose of the Data Set)

This dataset contains Move Orders and subinventory transfers Information.

Oracle E-Business Suite Table Names

```
mtl_material_transactions
```

ECC_SPEC_ID Columns

```
transaction_id,lot_number
```

ECC_SPEC_ID Concatenation Structure

```
to_char (1 || '-'||
 to_char (mtln.transaction_id)
|| '-'|| mtln.lot_number)
```

Data Set Key: `inv-trace-issues`

Data Set Name

Inventory Trace Issues

Description (Purpose of the Data Set)

This dataset contains Issue transactions Information.

Oracle E-Business Suite Table Names

```
mtl_material_transactions,mtl_transaction_lot_numbers
```

ECC_SPEC_ID Columns

```
transaction_id,lot_number
```

ECC_SPEC_ID Concatenation Structure

```
to_char (1||'-'|| to_char
(mtln.transaction_id)||'-'
||mtln.lot_number)
```

Data Set Key: `inv-trace-salesorders`

Data Set Name

Inventory Trace Sales Orders

Description (Purpose of the Data Set)

This dataset contains Sales Orders Information.

Oracle E-Business Suite Table Names

```
oe_order_headers_all,oe_order_lines_all,mtl_material_transactions
```

ECC_SPEC_ID Columns

```
header_id,line_id,transaction_id,lot_number,order_number,sales_order_id
```

ECC_SPEC_ID Concatenation Structure

```
to_char (so_txns.query_type|| '-'
 || so_txns.organization_id ||'-'
|| so_txns.transaction_id || '-'
||wdd.delivery_detail_id || '-'
|| so_txns.lot_number)
```

Data Set Key: `inv-physical-inventory`

Data Set Name

Inventory Physical Counting

Description (Purpose of the Data Set)

This dataset contains Physical Inventory Tags and adjustments Information.

Oracle E-Business Suite Table Names

```
mtl_physical_inventories,
mtl_physical_inventory_tags,
mtl_physical_adjustments
```

ECC_SPEC_ID Columns

```
physical_inventory_id,
tag_id,
adjustment_id
```

ECC_SPEC_ID Concatenation Structure

```
mpi.physical_inventory_id
                || '-'
                || mpa.inventory_item_id
                || '-'
                || mpa.adjustment_id
```

Data Set Key: `inv-move-orders`

Data Set Name

Inventory Move Orders

Description (Purpose of the Data Set)

This data set contains Move Orders Headers and Lines information.

Oracle E-Business Suite Table Names

```
mtl_txn_request_headers,
mtl_txn_request_lines
```

ECC_SPEC_ID Columns

```
header_id,line_id,
inventory_item_id,
organization_id
```

ECC_SPEC_ID Concatenation Structure

```
mtrl.line_id
        || '-'
        || mtrl.header_id
```

Data Set Key: `inv-move-order-allocations`

Data Set Name

Inventory Move Order Allocations

Description (Purpose of the Data Set)

This data set contains Move Order allocations information.

Oracle E-Business Suite Table Names

```
mtl_material_transactions_temp,mtl_transaction_lots_temp
```

ECC_SPEC_ID Columns

```
transaction_temp_id,lot_number,move_order_line_id
```

ECC_SPEC_ID Concatenation Structure

```
mmtt.transaction_temp_id
                || '-'
                || mtlt.lot_number
```

Order Management

Data Set Key: `ont-lines`

Data Set Name

Order Lines

Description (Purpose of the Data Set)

Outbound sales order lines.

Oracle E-Business Suite Table Names

`oe_order_lines_all`

ECC_SPEC_ID Columns

`line_id`

ECC_SPEC_ID Concatenation Structure

`line_id`

Data Set Key: `ont-headers`

Data Set Name

Orders

Description (Purpose of the Data Set)

Sales orders with one or more outbound sales order lines.

Oracle E-Business Suite Table Names

`oe_order_headers_all`

ECC_SPEC_ID Columns

`header_id`

ECC_SPEC_ID Concatenation Structure

`header_id`

Outsourced Manufacturing

Data Set Key: `jmf-subcon`

Data Set Name

- Order Status dashboard
- Financials dashboard

Description (Purpose of the Data Set)

This data set holds information about Subcontract Orders and Replenishment Orders details, Assembly and Component details, Adjustment details, and Quality details for both dashboards.

Oracle E-Business Suite Table Names

```
jmf_shikyu_replenishments (jsr)
jmf_subcontract_orders (jso)
jmf_shikyu_adjustments (jsadj)
jmf_shikyu_components (jsc)
qa_results (qr)
```

ECC_SPEC_ID Columns

```
jsr.replenishment_so_line_id
jsr.replenishment_so_header_id
jsr.replenishment_po_header_id
jsr.replenishment_po_line_id
jsr.shikyu_component_id
jso.subcontract_po_shipment_id
jsadj.adjustment_id
jso.subcontract_po_shipment_id
jsc.shikyu_component_id
jso.subcontract_po_shipment_id
qr.plan_id
qr.collection_id
qr.occurrence
```

ECC_SPEC_ID Concatenation Structure

```
jsr.replenishment_so_line_id
|| jsr.replenishment_so_header_id
|| jsr.replenishment_po_header_id
|| jsr.replenishment_po_line_id
|| jsr.shikyu_component_id
|| jso.subcontract_po_shipment_id
|| jsadj.adjustment_id
|| jso.subcontract_po_shipment_id
|| jsc.shikyu_component_id
|| jso.subcontract_po_shipment_id
|| qr.plan_id
|| qr.collection_id
|| qr.occurrence

UNION

jso.subcontract_po_shipment_id
|| jsc.shikyu_component_id
|| jsadj.adjustment_id
|| jso.subcontract_po_shipment_id
|| jsc.shikyu_component_id
|| jso.subcontract_po_shipment_id
|| qr.plan_id
|| qr.collection_id
|| qr.occurrence

UNION

jsr.replenishment_so_line_id
|| jsr.replenishment_so_header_id
|| jsr.replenishment_po_header_id
|| jsr.replenishment_po_line_id
|| jsr.shikyu_component_id
|| jso.subcontract_po_shipment_id
|| jsadj.adjustment_id
|| jso.subcontract_po_shipment_id
|| jsc.shikyu_component_id
|| jso.subcontract_po_shipment_id
|| qr.plan_id
|| qr.collection_id
|| qr.occurrence

UNION

jso.subcontract_po_shipment_id
|| jsc.shikyu_component_id
|| jsadj.adjustment_id
|| jso.subcontract_po_shipment_id
|| jsc.shikyu_component_id
|| jso.subcontract_po_shipment_id
|| qr.plan_id
|| qr.collection_id
|| qr.occurrence
```

Payables

Data Set Key: `ap-trx`

Data Set Name

Installments

Description (Purpose of the Data Set)

Open supplier invoices (all invoice types) with different validation status.

Oracle E-Business Suite Table Names

```
ap_payment_schedules_all
```

ECC_SPEC_ID Columns

```
invoice_id
payment_num
```

ECC_SPEC_ID Concatenation Structure

```
invoice_id||'_'||payment_num
```

Data Set Key: `ap-hold`

Data Set Name

Holds

Description (Purpose of the Data Set)

Unreleased holds on open invoices.

Oracle E-Business Suite Table Names

```
ap_holds_all
```

ECC_SPEC_ID Columns

```
invoice_id
hold_id
```

ECC_SPEC_ID Concatenation Structure

```
invoice_id || '_' || hold_id
```

Data Set Key: `ap-paid`

Data Set Name

Payments

Description (Purpose of the Data Set)

Payments in the last 7 days.

Oracle E-Business Suite Table Names

```
ap_payment_schedules_all
```

ECC_SPEC_ID Columns

```
invoice_payment_id
```

ECC_SPEC_ID Concatenation Structure

None.

Data Set Key: `ap-period-close`

Description (Purpose of the Data Set)

Data Set Name

AP Closing

Description (Purpose of the Data Set)

- Unaccounted invoices and payments
- Pending accounting events for invoices and payments (draft and invalid accounting)
- Accounted invoices and payments but not transferred yet to General Ledger

Oracle E-Business Suite Table Name: Unaccounted Invoices

ECC_SPEC_ID Columns

```
aia.invoice_id
aida.invoice_distribution_id
xe.event_id
xe.event_number
```

ECC_SPEC_ID Concatenation Structure

```
aia.invoice_id||'-
||aida.invoice_distribution_id||'-
||xe.event_id||'-'||xe.event_number
```

Oracle E-Business Suite Table Name: Accounted Invoices (not transferred to General Ledger)

ECC_SPEC_ID Columns

```
aia.invoice_id
aida.invoice_distribution_id
xe.event_id
xe.event_number
xah.ae_header_id
xal.ae_line_num
```

ECC_SPEC_ID Concatenation Structure

```
aia.invoice_id||'-'||aida.invoice_distribution_id||'-'||xe.
event_id||'-'||xe.event_number||'-'||xah.ae_header_id||'-'||xal.
ae_line_num
```

Oracle E-Business Suite Table Name: Unaccounted Payments

ECC_SPEC_ID Columns

```
ip.invoice_payment_id
xe.event_id
xe.event_number
```

ECC_SPEC_ID Concatenation Structure

```
ip.invoice_payment_id||'-'||xe.event_id||'-'||xe.event_number
```

Oracle E-Business Suite Table Name: Accounted Payments (not transferred to General Ledger)

ECC_SPEC_ID Columns

```
ip.invoice_payment_id
xe.event_id
xe.event_number
xah.ae_header_id
xal.ae_line_num
```

ECC_SPEC_ID Concatenation Structure

```
ip.invoice_payment_id||'-'||xe.event_id||'-'||xe.event_number||'-'||xah.
ae_header_id||'-'||xal.ae_line_num
```

Process Manufacturing

Data Set Key: gmo-batch

Data Set Name

Batch Dashboard

Description (Purpose of the Data Set)

This data set holds information about process manufacturing batch dashboard data.

ECC_SPEC_ID Concatenation Structure

The ecc_spec_id for this data set is the union of all other data sets except gmo-quality.

Data Set Key: gmo-batch-md

Data Set Name

Batch Details for Batch Dashboard

Description (Purpose of the Data Set)

This data set holds information about process manufacturing batch details for batch dashboard.

Oracle E-Business Suite Table Names

```
gme_batch_header(hdr)
```

ECC_SPEC_ID Columns

```
hdr.BATCH_ID
```

ECC_SPEC_ID Concatenation Structure

```
'A'||'-'|| TO_CHAR(hdr.BATCH_ID)
```

Data Set Key: gmo-batch-q

Data Set Name

Batch Details for Quality

Description (Purpose of the Data Set)

This data set holds information about process manufacturing batch details for quality dashboard.

Oracle E-Business Suite Table Names

```
gme_batch_header(hdr)
```

ECC_SPEC_ID Columns

```
hdr.BATCH_ID
```

ECC_SPEC_ID Concatenation Structure

```
'A'||'-'|| TO_CHAR(hdr.BATCH_ID)
```

Data Set Key: `gmo-material`

Data Set Name

Material Details for Batch

Description (Purpose of the Data Set)

This data set holds information about process manufacturing material details for the batch dashboard.

Oracle E-Business Suite Table Names

```
gme_material_details(batchdtl)
```

ECC_SPEC_ID Columns

```
batchdtl.BATCH_ID
batchdtl.material_detail_id
```

ECC_SPEC_ID Concatenation Structure

```
'B'||'-'||batchdtl.BATCH_ID ||'-'||batchdtl.material_detail_id
```

Data Set Key: `gmo-material-q`

Data Set Name

Material Details for Quality

Description (Purpose of the Data Set)

This data set holds information about process manufacturing material details for the quality dashboard.

Oracle E-Business Suite Table Names

```
gme_material_details(batchdtl)
```

ECC_SPEC_ID Columns

```
batchdtl.BATCH_ID
batchdtl.material_detail_id
```

ECC_SPEC_ID Concatenation Structure

```
'B'||'-'||batchdtl.BATCH_ID ||'-'||batchdtl.material_detail_id
```

Data Set Key: gmo-ncm

Data Set Name

Nonconformances Details

Description (Purpose of the Data Set)

This data set holds information about process manufacturing nonconformances details.

Oracle E-Business Suite Table Names

```
gme_batch_header(hdr)
qa_results (ncm)
```

ECC_SPEC_ID Columns

```
hdr.BATCH_ID
ncm.plan_id
ncm.collection_id
ncm.occurrence
```

ECC_SPEC_ID Concatenation Structure

```
'G' ||'-' || hdr.BATCH_ID ||'-' || ncm.plan_id ||'-' || ncm.
collection_id ||'-' || ncm.occurrence
```

Data Set Key: gmo-quality

Data Set Name

Production Quality

Description (Purpose of the Data Set)

This data set holds information about process manufacturing production quality dashboard data.

ECC_SPEC_ID Concatenation Structure

The `ecc_spec_id` for this data set is the union of all other data sets except `gmo-batch` .

Data Set Key: gmo-res

Data Set Name

Resource Details for Batch

Description (Purpose of the Data Set)

This data set holds information about process manufacturing resource details for the batch dashboard.

Oracle E-Business Suite Table Names

```
GME_BATCH_STEP_RESOURCES (res)
```

ECC_SPEC_ID Columns

```
res.BATCH_ID
res.batchstep_id
res.batchstep_activity_id
res.BATCHSTEP_RESOURCE_ID
pp.PROCESS_PARAM_ID
```

ECC_SPEC_ID Concatenation Structure

```
'F'||'-'||res.BATCH_ID ||'-'||res.batchstep_id||'-'||res.
batchstep_activity_id||'-'|| res.BATCHSTEP_RESOURCE_ID. UNION
'C'||'-'||res.BATCH_ID ||'-'|| res.batchstep_id||'-'||res.
batchstep_activity_id||'-'|| res.BATCHSTEP_RESOURCE_ID||'-'||pp.
PROCESS_PARAM_ID
```

Data Set Key: `gmo-res-q`

Data Set Name

Resource Details for Quality

Description (Purpose of the Data Set)

This data set holds information about process manufacturing resource details for the quality dashboard.

Oracle E-Business Suite Table Names

```
GME_BATCH_STEP_RESOURCES (res)
```

ECC_SPEC_ID Columns

```
res.BATCH_ID
res.batchstep_id
res.batchstep_activity_id
res.BATCHSTEP_RESOURCE_ID
pp.PROCESS_PARAM_ID
```

ECC_SPEC_ID Concatenation Structure

```
'F'||'-'||res.BATCH_ID ||'-'||res.batchstep_id||'-'||res.
batchstep_activity_id||'-'|| res.BATCHSTEP_RESOURCE_ID. UNION
'C'||'-'||res.BATCH_ID ||'-'|| res.batchstep_id||'-'||res.
batchstep_activity_id||'-'|| res.BATCHSTEP_RESOURCE_ID||'-'||pp.
PROCESS_PARAM_ID
```

Data Set Key: gmo-sample

Data Set Name

Sample Details

Description (Purpose of the Data Set)

This data set holds information about process manufacturing sample details.

Oracle E-Business Suite Table Names

```
gmd_samples (s)
gmd_spec_results (sr)
gmd_results (r)
```

ECC_SPEC_ID Columns

```
s.BATCH_ID
s.sample_id
sr.event_spec_disp_id
r.result_id
```

ECC_SPEC_ID Concatenation Structure

```
'H'||'-'||s.BATCH_ID||'-'|| s.sample_id||'-'|| sr.
event_spec_disp_id||'-'|| r.result_id
```

Data Set Key: gmo-so

Data Set Name

Sales Order Details

Description (Purpose of the Data Set)

This data set holds information about process manufacturing sales order details.

Oracle E-Business Suite Table Names

```
gme_batch_header (hdr)
mtl_reservations (rsrv)
```

ECC_SPEC_ID Columns

```
hdr.BATCH_ID
rsrv.reservation_id
```

ECC_SPEC_ID Concatenation Structure

```
'I'||'-'||hdr.BATCH_ID ||'-'|| rsrv.reservation_id
```

Data Set Key: gmo-steps

Data Set Name

Step Details for Batch

Description (Purpose of the Data Set)

This data set holds information about process manufacturing step details for the batch dashboard.

Oracle E-Business Suite Table Names

```
GME_BATCH_STEPS (step)
```

ECC_SPEC_ID Columns

```
step.BATCH_ID
step.batchstep_id
```

ECC_SPEC_ID Concatenation Structure

```
'D'||'-'||step.BATCH_ID ||'-'||step.batchstep_id
```

Data Set Key: gmo-steps-q

Data Set Name

Step Details for Quality

Description (Purpose of the Data Set)

This data set holds information about process manufacturing step details for the quality dashboard.

Oracle E-Business Suite Table Names

```
GME_BATCH_STEPS (step)
```

ECC_SPEC_ID Columns

```
step.BATCH_ID
step.batchstep_id
```

ECC_SPEC_ID Concatenation Structure

```
'D'||'-'||step.BATCH_ID ||'-'||step.batchstep_id
```

Data Set Key: gmo-test

Data Set Name

Test Details

Description (Purpose of the Data Set)

This data set holds information about process manufacturing test details.

Oracle E-Business Suite Table Names

```
gmd_samples (s)
gmd_spec_results (sr)
gmd_results (r)
```

ECC_SPEC_ID Columns

```
s.BATCH_ID
s.sample_id
sr.event_spec_disp_id
r.result_id
```

ECC_SPEC_ID Concatenation Structure

```
'H'||'-'||s.BATCH_ID||'-'|| s.sample_id||'-'|| sr.
event_spec_disp_id||'-'|| r.result_id
```

Data Set Key: gmo-exp

Data Set Name

Expiry Lot Details

Description (Purpose of the Data Set)

This data set holds information about process manufacturing expiry lot details.

Oracle E-Business Suite Table Names

```
mtl_lot_numbers(mln)
```

ECC_SPEC_ID Columns

```
mln.organization_id
mln.inventory_item_id
mln.lot_number
```

ECC_SPEC_ID Concatenation Structure

```
'K'||'-'||mln.organization_id ||mln.inventory_item_id ||mln.lot_number
```

Data Set Key: gmo-ospdetails

Data Set Name

Outside Processing Batch

Description (Purpose of the Data Set)

This data set holds information about process manufacturing outside processing operations.

Oracle E-Business Suite Table Names

```
gme_batch_header (gbh)
gme_batch_steps (gbs)
cr_rsrc_mst_b (crm)
po_distributions_all pd
```

ECC_SPEC_ID Columns

```
gbh.batch_id,
gbs.batchstep_id,
crm.resources,
gbh.organization_id,
pd.rec_key,
```

ECC_SPEC_ID Concatenation Structure

```
gbh.batch_id || ''-'' || gbs.batchstep_id || ''-'' || crm.resources ||
''-'' || gbh.organization_id || ''-'' || pd.rec_key || ''-OSP''
```

Procurement

Data Set Key: `po-pcc-agreements`

Data Set Name

Agreements

Description (Purpose of the Data Set)

This gives information on agreement details.

Oracle E-Business Suite Table Names

```
po_headers_all
po_lines_all
okc_deliverables
```

ECC_SPEC_ID Columns

```
poh.po_header_id
pla.po_line_id
del.deliverable_id
```

ECC_SPEC_ID Concatenation Structure

```
poh.po_header_id||'-'||pla.po_line_id||'-'||del.deliverable_id
```

Data Set Key: `po-pcc-requisitions`

Data Set Name

Requisitions

Description (Purpose of the Data Set)

This gives information on requisitions and their statuses.

Oracle E-Business Suite Table Names

```
po_requisition_headers_all
po_requisition_lines_all
po_req_distributions_all
```

ECC_SPEC_ID Columns

```
req_header.requisition_header_id
req_line.requisition_line_id
req_dist.distribution_Id
```

ECC_SPEC_ID Concatenation Structure

```
req_header.requisition_header_id || ''-'' || req_line.
requisition_line_id || ''-'' || req_dist.distribution_Id
```

Data Set Key: `po-pcc-orders`

Data Set Name

Purchase Orders

Description (Purpose of the Data Set)

This gives detailed information on purchase orders and their statuses.

Oracle E-Business Suite Table Names

```
po_headers_all
po_lines_all
po_line_locations_all
po_distributions_all
okc_deliverables
```

ECC_SPEC_ID Columns

```
poh.po_header_id
pla.po_line_id
plla.line_location_id
pda.PO_DISTRIBUTION_ID
del.deliverable_id
```

ECC_SPEC_ID Concatenation Structure

```
poh.po_header_id||'-'||pla.po_line_id||'-'||plla.
line_location_id||'-'||pda.PO_DISTRIBUTION_ID || '-' ||del.
deliverable_id
```

Data Set Key: `po-pcc-orders-metric`

Data Set Name

Purchase Order Metrics

Description (Purpose of the Data Set)

This data set provides information on invoices on hold for the purchase orders.

Oracle E-Business Suite Table Names

```
ap_holds_all
```

ECC_SPEC_ID Columns

```
aha.INVOICE_ID
aha.LINE_LOCATION_ID
aha.HOLD_LOOKUP_CODE
```

ECC_SPEC_ID Concatenation Structure

```
aha.INVOICE_ID ||'-'|| aha.LINE_LOCATION_ID ||'-'||REPLACE(aha.
HOLD_LOOKUP_CODE,' ','_')
```

Data Set Key: `po-pcc-orders-prccycletimegraph`

Data Set Name

Procurement Cycle Time Graph

Description (Purpose of the Data Set)

This provides information on procurement action lead time (PALT).

Oracle E-Business Suite Table Names

```
po_headers_all
```

ECC_SPEC_ID Columns

```
poh.po_header_id
```

ECC_SPEC_ID Concatenation Structure

```
poh.po_header_id
```

Data Set Key: `po-pcc-receipts`

Data Set Name

PCC Receipts

Description (Purpose of the Data Set)

The receipts related to the purchase orders.

Oracle E-Business Suite Table Names

```
rcv_shipment_headers
rcv_shipment_lines
rcv_transactions
po_lines_all
po_headers_all
```

ECC_SPEC_ID Columns

```
rcv_shipment_headers.shipment_header_id
rcv_shipment_lines.shipment_line_id|
rcv_transactions.transaction_id
```

ECC_SPEC_ID Concatenation Structure

```
'RECEIPT'
||'-'
||rcv_shipment_headers.shipment_header_id
||'-'
||rcv_shipment_lines.shipment_line_id
||'-'
||rcv_transactions.transaction_id
```

Data Set Key: `po-pcc-invoices`

Data Set Name

PCC Invoices

Description (Purpose of the Data Set)

The invoices related to the purchase orders.

Oracle E-Business Suite Table Names

```
ap_invoices_all
ap_invoice_distributions_all
po_distributions_all
po_headers_all
po_lines_all
```

ECC_SPEC_ID Columns

```
ap_invoices_all.INVOICE_ID
ap_invoice_distributions_all.INVOICE_DISTRIBUTION_ID
```

ECC_SPEC_ID Concatenation Structure

```
'INVOICE'
||'-'
||ap_invoices_all.INVOICE_ID
||'-'
||ap_invoice_distributions_all.INVOICE_DISTRIBUTION_ID
```

Product Information Management

Data Set Key: `inv-item-grain`

Data Set Name

Organization Items

Description (Purpose of the Data Set)

This data set brings in information on Items and their various attributes.

Oracle E-Business Suite Table Names

```
mtl_system_items_b
```

ECC_SPEC_ID Columns

```
inventory_item_id,organization_id
```

ECC_SPEC_ID Concatenation Structure

```
inventory_item_id||'_'||organization_id
```

Data Set Key: `inv-item-categories`

Data Set Name

Item Category Sets

Description (Purpose of the Data Set)

This data set provides information on the item's category sets and categories.

Oracle E-Business Suite Table Names

```
inventory_item_id, organization_id, category_set_id, category_id
```

ECC_SPEC_ID Columns

```
inventory_item_id||'_'||organization_id||'_'||categry_set_id||'_'||categ
ory_id
```

ECC_SPEC_ID Concatenation Structure

None.

Data Set Key: `inv-item-functions`

Data Set Name

Item Functions

Description (Purpose of the Data Set)

This data set is used to determine the number of items against each item function, such as Manufacturing, Stockable, and so on.

Oracle E-Business Suite Table Names

```
mtl_system_items_b
```

ECC_SPEC_ID Columns

```
inventory_item_id,organization_id
```

ECC_SPEC_ID Concatenation Structure

```
inventory_item_id||'_'||organization_id
```

Data Set Key: `inv-item-suppliers`

Data Set Name

Approved Supplier Items

Description (Purpose of the Data Set)

The data set shows the approved suppliers for an item.

Oracle E-Business Suite Table Names

```
mtl_system_items_b
```

ECC_SPEC_ID Columns

```
inventory_item_id,organization_id,vendor_name
```

ECC_SPEC_ID Concatenation Structure

```
 inventory_item_id||'_'||organization_id||'_'||vendor_name
```

Data Set Key: `inv-item-revisions`

Data Set Name

Item Revisions

Description (Purpose of the Data Set)

This data set brings in an item and all the revisions present for that item.

Oracle E-Business Suite Table Names

```
mtl_system_items_b,
mtl_item_revisions_b
```

ECC_SPEC_ID Columns

```
inventory_item_id,organization_id,revision_id
```

ECC_SPEC_ID Concatenation Structure

```
inventory_item_id||'_'||organization_id||'_'||revision_id
```

Data Set Key: `inv-item-manufactures`

Data Set Name

Manufactured Items

Description (Purpose of the Data Set)

This data set shows the manufacturer and the manufacturer part number for each item.

Oracle E-Business Suite Table Names

```
 mtl_system_items_b,
mtl_mfg_part_numbers
```

ECC_SPEC_ID Columns

```
inventory_item_id, organization_id, manufacturer_id, mfg_part_num
```

ECC_SPEC_ID Concatenation Structure

```
inventory_item_id||'_'||organization_id||'_'||manufacturer_id||'_'||mfg_
part_num
```

Data Set Key: `inv-cross-ref`

Data Set Name

Item Cross-Reference

Description (Purpose of the Data Set)

This data set shows all the cross-references and their types of an item.

Oracle E-Business Suite Table Names

```
mtl_system_items_b,
mtl_cross_refernces_b
```

ECC_SPEC_ID Columns

```
 inventory_item_id,organization_id,cross_reference_id
```

ECC_SPEC_ID Concatenation Structure

```
 inventory_item_id||'_'||organization_id||'_'||cross_reference_id
```

Data Set Key: `inv-related-item`

Data Set Name

Related Items

Description (Purpose of the Data Set)

This data set shows all the items related to a specific item and the relationship type.

Oracle E-Business Suite Table Names

```
 mtl_system_items_b,
mtl_related_items
```

ECC_SPEC_ID Columns

```
 inventory_item_id,organization_id,related_item_id,relationship_type_id
```

ECC_SPEC_ID Concatenation Structure

```
inventory_item_id||'_'||organization_id||'_'||related_item_id||'_'||rela
tionship_type_id
```

Project Manufacturing (PJM)

Data Set Key: `pjm-work-orders`

Data Set Name

Work orders

Description (Purpose of the Data Set)

This data set contains work order details for the work order dashboard

Oracle E-Business Suite Table Names

```
wip_discrete_jobs
wip_entities
wip_operations
wip_requirement_operations
bom_standard_operations
```

ECC_SPEC_ID Columns

```
project_id,
task_id,
wip_entity_id,
organization_id,
operation_seq_num,
inventory_item_id,
standard_operation_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_WO'|| '-'|| wdj.project_id|| '-'
|| nvl(to_char(wdj.task_id), 'NO_TASK')|| '-'
|| we.wip_entity_id|| '-'|| wdj.organization_id|| '-'||nvl(to_char(wo.
operation_seq_num),'NO_OP')|| '-'|| nvl(to_char(wro.inventory_item_id),
'NO_COMP')|| '-'||nvl(to_char(bso.standard_operation_id), 'NO_STDOP')
```

Data Set Key: `pjm-line-schedules`

Data Set Name

Line Schedules

Description (Purpose of the Data Set)

This data set contains line schedule details in the work order dashboard

Oracle E-Business Suite Table Names

```
pa_projects_all,
WIP_FLOW_SCHEDULES,
WIP_LINES
```

ECC_SPEC_ID Columns

```
project_id,
task_id,
line_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_LS' || '-' || PROJECT_ID || '-' || NVL(TO_CHAR(TASK_ID),
'NO_TASK') || '-' || LINE_ID
```

Data Set Key: `pjm-sales-orders`

Data Set Name

Sales Orders

Description (Purpose of the Data Set)

This data set contains sales order details in the Sales Order dashboard

Oracle E-Business Suite Table Names

```
oe_order_lines_all,
oe_order_headers_all
```

ECC_SPEC_ID Columns

```
project_id,
task_id,
header_id,
line_id
```

ECC_SPEC_ID Concatenation Structure

```
 'PROJ_SO' || '-' || ool.project_id || '-' || NVL( TO_CHAR(ool.task_id),
'NO_TASK' ) || '-' || ooh.header_id || '-' || ool.line_id
```

Data Set Key: `pjm-inv-balance`

Data Set Name

Onhand balance

Description (Purpose of the Data Set)

This data set contains inventory balance details in the Inventory dashboard

Oracle E-Business Suite Table Names

```
mtl_item_locations,
mtl_onhand_quantities_detail,
mtl_item_locations_kfv,
mtl_system_items_kfv
```

ECC_SPEC_ID Columns

```
project_id,
task_id,
organization_id,
inventory_item_id,
concatenated_segments,
primary_uom_code
```

ECC_SPEC_ID Concatenation Structure

```
 'PROJ_INV_BAL' || '-' || loc.project_id || '-' || NVL( TO_CHAR(loc.
task_id), 'NO_TASK' ) || '-' || qty.organization_id || '-' || qty.
inventory_item_id || '-' || phyloc.concatenated_segments || '-' || item.
primary_uom_code
```

Data Set Key: `pjm-onhand-values`

Data Set Name

Onhand values

Description (Purpose of the Data Set)

This data set contains details for on-hand values in the Inventory dashboard

Oracle E-Business Suite Table Names

```
pjm_project_parameters,
mtl_onhand_quantities,
cst_quantity_layers
```

ECC_SPEC_ID Columns

```
project_id,
organization_id,
inventory_item_id,
cost_group_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_INV_VAL' || '-' || para.project_id || '-' || para.organization_id
|| '-' || qty.inventory_item_id || '-' || layer.cost_group_id
```

Data Set Key: `pjm-projects`

Data Set Name

Projects

Description (Purpose of the Data Set)

This data set contains project-related information in the Project dashboard

Oracle E-Business Suite Table Names

```
pa_projects_all
```

ECC_SPEC_ID Columns

```
project_id
```

ECC_SPEC_ID Concatenation Structure

```
PROJ-' || pp.project_id
```

Data Set Key: `pjm-tasks`

Data Set Name

Tasks

Description (Purpose of the Data Set)

This data set contains task-related information in the Project dashboard

Oracle E-Business Suite Table Names

```
pa_projects_all,
pa_tasks
```

ECC_SPEC_ID Columns

```
project_id,
task_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_TASK-' || pt.project_id || '-' || pt.task_id
```

Data Set Key: `pjm-expenditures`

Data Set Name

Expenditures

Description (Purpose of the Data Set)

This data set contains expenditure-related information in the Project dashboard

Oracle E-Business Suite Table Names

```
pa_projects_all,
pa_tasks,
pa_expenditure_items_all
```

ECC_SPEC_ID Columns

```
project_id,
task_id,
expenditure_item_id
```

ECC_SPEC_ID Concatenation Structure

```
 'PROJ_EXP-' || pp.project_id || '-' || NVL( TO_CHAR(pt.task_id),
'NO_TASK' ) || '-' || TO_CHAR(pei.expenditure_item_id)
```

Data Set Key: `pjm-commitments`

Data Set Name

Commitments

Description (Purpose of the Data Set)

This data set contains commitment-related information in the Project dashboard

Oracle E-Business Suite Table Names

```
pa_projects_all,
pa_tasks,
pa_commitment_txns
```

ECC_SPEC_ID Columns

```
project_id,
task_id,
cmt_line_id
```

ECC_SPEC_ID Concatenation Structure

```
 'PROJ_CMT-' || pp.project_id || '-' || NVL( TO_CHAR(pt.task_id),
'NO_TASK' ) || '-' || pct.cmt_line_id
```

Data Set Key: `pjm-exceptions-overdues`

Data Set Name

Exceptions and Overdues

Description (Purpose of the Data Set)

This dataset contains exception related information in Overview dashboard

Oracle E-Business Suite Table Names: `wip_discrete_jobs`

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
WIP_ENTITY_ID
```

ECC_SPEC_ID Concatenation Structure

```
 'PROJ_WIP_SCHEDULE' || '-' || WDJ.PROJECT_ID || '-' || NVL( TO_CHAR
(WDJ.TASK_ID), 'NO_TASK' ) || '-' || WDJ.WIP_ENTITY_ID
```

Oracle E-Business Suite Table Names: `wip_discrete_jobs`

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
WIP_ENTITY_ID
```

ECC_SPEC_ID Concatenation Structure

```
 'PROJ_WIP_OVERDUE' || '-' || WDJ.PROJECT_ID || '-' || NVL( TO_CHAR(WDJ.
TASK_ID), 'NO_TASK' ) || '-' || TO_CHAR(WDJ.WIP_ENTITY_ID)
```

Oracle E-Business Suite Table Names: `oe_order_lines_all`

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
HEADER_ID,
LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_SO_SCHEDULE' || '-' || OOL.PROJECT_ID || '-' || NVL( TO_CHAR(OOL.
TASK_ID), 'NO_TASK' ) || '-' || OOL.HEADER_ID || '-' || OOL.LINE_ID
```

Oracle E-Business Suite Table Names: oe_order_lines_all, oe_order_headers_all

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
HEADER_ID,
LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_SO_OVERDUE' || '-' || OOL.PROJECT_ID || '-' || NVL( TO_CHAR(OOL.
task_id), 'NO_TASK' ) || '-' || OOH.HEADER_ID || '-' || OOL.LINE_ID
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
HEADER_ID,
LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_SCHEDULE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-PO-' || PL.PO_HEADER_ID || '-' || PL.
PO_LINE_ID
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
HEADER_ID,
LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_SCHEDULE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-BR-' || PL.PO_HEADER_ID || '-' || PL.
PO_LINE_ID
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, po_requisition_lines_all

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
REQUISITION_HEADER_ID,
requisition_line_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_SCHEDULE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-REQ-' || prl.REQUISITION_HEADER_ID || '-' ||
prl.requisition_line_id
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
po_header_id,
po_line_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_SCHEDULE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-RFQ-' || pl.po_header_id || '-' || pl.
po_line_id
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
po_header_id,
PO_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_SCHEDULE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-QUO-' || pl.po_header_id || '-' || pl.
po_line_id
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
PO_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_OVERDUE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-PO_' || TO_CHAR(PL.PO_LINE_ID)
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_OVERDUE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-BR_' || TO_CHAR(PL.PO_LINE_ID)
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, po_requisition_lines_all

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
requisition_line_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_OVERDUE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-REQ_' || TO_CHAR(PRL.REQUISITION_LINE_ID)
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
PO_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_OVERDUE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-RFQ_' || TO_CHAR(PL.PO_LINE_ID)
```

Oracle E-Business Suite Table Names: PA_PROJECTS_ALL, PA_TASKS, PO_LINES_ALL

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
PO_LINE_ID
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PA_OVERDUE' || '-' || PP.PROJECT_ID || '-' || NVL( TO_CHAR(PT.
TASK_ID), 'NO_TASK' ) || '-QT_' || TO_CHAR(PL.PO_LINE_ID)
```

Data Set Key: pjm-procurement

Data Set Name

Procurement Activities

Description (Purpose of the Data Set)

This data set contains Procurement-related information in the Procurement dashboard.

Oracle E-Business Suite Table Names: po_distributions_all, po_headers_all, po_lines_all, po_line_locations_all

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
po_header_id,
po_line_id,
line_location_id,
po_distribution_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_PO' || '-' || dist.project_id || '-' || NVL( TO_CHAR(dist.
task_id), 'NO_TASK' ) || '-' || poh.po_header_id || '-' || pol.
po_line_id || '-' || poll.line_location_id || '-' || dist.
po_distribution_id
```

Oracle E-Business Suite Table Names: po_req_distributions_all, po_requisition_headers_all, po_requisition_lines_all

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
requisition_header_id,
requisition_line_id,
distribution_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_REQ' || '-' || dist.project_id || '-' || NVL( TO_CHAR(dist.
task_id), 'NO_TASK' ) || '-' || prh.requisition_header_id || '-' ||
line.requisition_line_id || '-' || dist.distribution_id
```

Oracle E-Business Suite Table Names: po_distributions_all, po_releases_all, po_lines_all, po_line_locations_all

ECC_SPEC_ID Columns

```
PROJECT_ID,
TASK_ID,
po_header_id,
po_release_id,
line_num,
shipment_num,
distribution_num
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_BR' || '-' || dist.project_id || '-' || NVL( TO_CHAR(dist.
task_id), 'NO_TASK' ) || '-' || por.po_header_id || '-' || por.
po_release_id || '-' || pol.line_num || '-' || poll.shipment_num || '-'
|| dist.distribution_num
```

Oracle E-Business Suite Table Names: `po_lines_all, po_headers_all`

ECC_SPEC_ID Columns

```
project_id,
task_id,
po_header_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_QUOT' || '-' || line.project_id || '-' || NVL( TO_CHAR(line.
task_id), 'NO_TASK' ) || '-' || poh.po_header_id || '-' || poh1.
po_header_id
```

Oracle E-Business Suite Table Names: `po_lines_all, po_headers_all`

ECC_SPEC_ID Columns

```
project_id,
task_id,
po_header_id,
po_line_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_RFQ'
|| '-'
|| line.project_id
|| '-'
|| NVL( TO_CHAR(line.task_id), 'NO_TASK' )
|| '-'
|| poh.po_header_id
|| '-'
|| line.po_line_id
```

Data Set Key: `pjm-project-contracts`

Data Set Name

PJM Project Contracts

Description (Purpose of the Data Set)

This Dataset contains Project Contract information in the Project Contracts Dashboard.

Oracle E-Business Suite Table Names

```
oke_k_headers,
oke_k_lines,
oke_k_deliverables_b
```

ECC_SPEC_ID Columns

```
k_header_id,
k_line_id,
deliverable_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_CONTRACT'
|| '-'
|| okeh.k_header_id
|| '-'
|| okel.k_line_id
|| '-'
|| oked.deliverable_id
```

Data Set Key: `pjm-cost-activities`

Data Set Name

PJM Cost Activities

Description (Purpose of the Data Set)

This Dataset contains Cost Activities information in the Cost Activities Dashboard.

Oracle E-Business Suite Table Names

```
wip_discrete_jobs (WIP Summary)
```

ECC_SPEC_ID Columns

```
project_id,
wip_entity_id
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ_CA'
|| '-'
|| wdj.project_id
|| '-WIPVS-'
|| wdj.wip_entity_id
```

Projects

Data Set Key: `pa-ds-cost-eicdl`

Data Set Name

Projects Costing: Transactions

Description (Purpose of the Data Set)

Expenditure level information for the transactions charged on Projects.

Oracle E-Business Suite Table Names

```
pa_exp_item_mls_v
pa_cost_distribution_lines_all
pa_expenditure_groups_all
pa_system_linkages
gl_code_combinations_kfv
hr_all_organization_units
pa_cc_tp_schedules
xla_events
xla_ae_headers
xla_accounting_errors
```

ECC_SPEC_ID Columns

```
ei.expenditure_item_id
cdl.line_num
cdl.line_type
```

ECC_SPEC_ID Concatenation Structure

```
''PROJ-''|| TO_CHAR(ei.expenditure_item_id)|| ''-''||
TO_CHAR(cdl.line_num)|| ''-''||
cdl.line_type
```

Data Set Key: `pa-ds-cost-asset`

Data Set Name

Projects Costing: Assets

Description (Purpose of the Data Set)

Assets related information on the Capital Projects.

Oracle E-Business Suite Table Names

```
pa_implementations_all
gl_sets_of_books
pa_project_types_all
pa_projects_all
pa_project_assets_all
pa_project_asset_assignments
pa_project_asset_lines_all
pa_project_asset_line_details
hr_all_organization_units_tl
```

ECC_SPEC_ID Columns

```
ppal.project_asset_id
pp.project_id
ppal.project_asset_line_id
ppal.task_id
ppal.project_asset_line_detail_id
ppald.proj_asset_line_dtl_uniq_id
```

ECC_SPEC_ID Concatenation Structure

```
''PROJ-' ||
TO_CHAR(NVL(ppal.project_asset_id,
pp.project_id)) || '-' ||
TO_CHAR(NVL(ppal.project_asset_line_id,
pp.project_id)) || '-' ||
TO_CHAR(NVL(ppal.task_id, 99999)) || '-' ||
TO_CHAR(NVL(ppal.project_asset_line_detail_id, pp.project_id)) || '-' ||

TO_CHAR(NVL(ppald.proj_asset_line_dtl_uniq_id, pp.project_id))
```

Data Set Key: `pa-ds-cost-source-imp`

Data Set Name

Projects Costing: Transaction Source

Description (Purpose of the Data Set)

Transaction details charged outside Oracle Projects. These are details from approved documents pending interface or transactions pending import to Oracle Projects.

Oracle E-Business Suite Table Names

```
pa_transaction_interface_all
pa_transaction_sources
pa_project_types_all
pa_projects_all
pa_tasks
hr_all_organization_units_tl
gl_code_combinations_kfv
hr_locations
po_vendors
po_headers_all
po_lines_all
po_distributions_all
rcv_shipment_headers
rcv_transactions
rcv_shipment_lines
rcv_receiving_sub_ledger
rcv_accounting_event_types
rcv_accounting_events
ap_invoices_all
ap_invoice_lines_all
ap_invoice_distributions_all
ap_payment_hist_dists
ap_invoice_payments_all
ap_prepay_app_dists
```

ECC_SPEC_ID Columns

```
ps.transaction_id
ps.distribution_id
ts.transaction_source
```

ECC_SPEC_ID Concatenation Structure

```
'PROJ-' ||
TO_CHAR(ps.transaction_id) || '-' ||
TO_CHAR(ps.distribution_id) || '-' ||
TO_CHAR(ts.transaction_source),
```

Data Set Key: `pa-ds-cost-psi-summary`

Data Set Name

Projects Costing: PSI Summary

Description (Purpose of the Data Set)

Transaction and Budget related details at all levels (Project, Top Task, Child Tasks and Leaf Tasks) as per summarized data available on Projects.

Oracle E-Business Suite Table Names

```
pa_projects_all
pa_project_accum_headers
pa_project_accum_budgets
pa_project_accum_budgets
pa_project_accum_actuals
pa_project_accum_commitments
pa_project_types_all
pa_project_statuses
hr_all_organization_units_tl
pa_projects_all
pa_tasks
per_all_people_f
pa_ind_rate_schedules
```

ECC_SPEC_ID Columns

```
accum.org_id
accum.project_id
accum.task_id
```

ECC_SPEC_ID Concatenation Structure

```
'PSI_PROJ_TASK_' ||
TO_CHAR(accum.org_id) || '-' ||
TO_CHAR(accum.project_id) || '-' ||
TO_CHAR(accum.task_id)
```

Data Set Key: `pa-ds-cost-bcbalances`

Data Set Name

Projects Costing: Budget Balances

Description (Purpose of the Data Set)

Project and Task level Budget balances for Budgetary Control Enabled Projects. This helps to track Budget, Commitments and Actuals period-wise.

Oracle E-Business Suite Table Names

```
hr_all_organization_units_tl
per_all_people_f
pa_bc_packets
pa_bc_balances
pa_periods_all
pa_budget_entry_methods
pa_budget_versions
gl_period_statuses
pa_budgetary_control_options
pa_budget_types
pa_tasks
pa_project_types_all
pa_project_statuses
pa_projects_all
```

ECC_SPEC_ID Columns

```
bgt.project_id
bgt.encumb_type
bgt.task_id
bgt.top_task_id
bgt.parent_member_id
bgt.resource_list_member_id
bgt.start_date
bgt.end_date
```

ECC_SPEC_ID Concatenation Structure

```
'PBC-' || TO_CHAR(bgt.project_id) || '-' ||
TO_CHAR(bgt.encumb_type) || '-' ||
TO_CHAR(bgt.task_id) || '-' ||
TO_CHAR(bgt.top_task_id) || '-' ||
TO_CHAR(bgt.parent_member_id) || '-' ||
TO_CHAR(bgt.resource_list_member_id) || '-' ||
TO_CHAR(bgt.start_date,'DD-MON-YYYY') || '-' ||
TO_CHAR(bgt.end_date,'DD-MON-YYYY')
```

Data Set Key: `pa-ds-cost-bcresults`

Data Set Name

Projects Costing: Budget Results

Description (Purpose of the Data Set)

Details of transaction funds check results and related document rejections for Budgetary Control Enabled Projects.

Oracle E-Business Suite Table Names

```
pa_expenditure_types
po_headers_all
po_lines_all
po_distributions_all
pa_bc_commitments_all
ap_invoices_all
ap_invoice_lines_all
ap_invoice_distributions_all
po_vendors
gl_ledgers
pa_implementations_all
pa_projects_all
pa_tasks
po_req_distributions_all
po_requisition_lines_all
po_requisition_headers_all
pa_bc_commitments_all
pa_budget_versions
pa_budgetary_control_options
pa_expenditures_all
pa_expenditure_items_all
pa_cost_distribution_lines_all
pa_bc_packets
pa_budget_types
pa_project_statuses
pa_periods_all
hr_all_organization_units_tl
```

ECC_SPEC_ID Columns

```
det.record_identifier
det.project_id
det.task_id
det.bc_commitment_ei_id
det.burden_cost_flag
det.document_type
det.document_header_id
det.document_distribution_id
```

ECC_SPEC_ID Concatenation Structure

```
'PBC-'|| det.record_identifier||
TO_CHAR(det.project_id)|| '-' ||
TO_CHAR(det.task_id)|| '-' ||
TO_CHAR(det.bc_commitment_ei_id)|| '-' ||
TO_CHAR(det.burden_cost_flag)|| '-' ||
TO_CHAR(det.document_type)|| '-' ||
TO_CHAR(det.document_header_id)|| '-' ||
TO_CHAR(det.document_distribution_id)
```

Data Set Key: `pa-ds-bill-funding`

Data Set Name

Projects Billing: Funding

Description (Purpose of the Data Set)

Stores summary funding amounts allocated from agreements to projects and tasks for

Contract Projects.

Oracle E-Business Suite Table Names

```
pa_projects_all
pa_project_statuses
pa_project_types_all
pa_budget_versions
pa_budget_types
pa_budget_entry_methods
pa_implementations_all
pa_summary_project_fundings
pa_tasks
hz_parties
hz_cust_accounts
pa_agreements_all
pa_summary_project_retn
hr_all_organization_units_tl
```

ECC_SPEC_ID Columns

```
p.project_id
spf.task_id
spf.agreement_id
```

ECC_SPEC_ID Concatenation Structure

```
'FUND-'
|| TO_CHAR(p.project_id)
|| '-' || TO_CHAR(nvl(spf.task_id, 0))
|| '-' || TO_CHAR(spf.agreement_id)
```

Data Set Key: `pa-ds-bill-budget`

Data Set Name

Projects Billing: Budget

Description (Purpose of the Data Set)

Summarized budget line data for the Project and TopTask combination of the latest baseline budget version for Contract Projects.

Oracle E-Business Suite Table Names

```
pa_projects_all
pa_project_statuses
pa_project_types_all
pa_budget_versions
pa_budget_types
pa_budget_entry_methods
pa_implementations_all
pa_budget_lines
pa_resource_assignments
pa_tasks
hr_all_organization_units_tl
gl_period_statuses
```

ECC_SPEC_ID Columns

```
p.project_id
pra.task_id
pbl.BUDGET_LINE_ID
pbl.start_date
pbl.end_date
```

ECC_SPEC_ID Concatenation Structure

```
'BGT-'
|| TO_CHAR(p.project_id)
|| '-' || TO_CHAR(nvl(pra.task_id, 0))
|| '-' || TO_CHAR(pbl.BUDGET_LINE_ID)
|| '-' || TO_CHAR(pbl.start_date,'DD-MON-YYYY')
|| '-' || TO_CHAR(pbl.end_date,'DD-MON-YYYY')
```

Data Set Key: `pa-ds-bill-revenueinvoice`

Data Set Name

Projects Billing: Revenue, Invoice, and Distributions

Description (Purpose of the Data Set)

Stores draft revenue, draft invoice and their distribution details for the transactions (both events and expenditures) contributing to either revenue or invoice, or both in Contract Projects.

Oracle E-Business Suite Table Names

```
pa_projects_all
pa_project_statues
pa_project_types_all
pa_budget_versions
pa_budget_types
pa_budget_entry_methods
pa_implementations_all
pa_tasks
pa_draft_invoices_all
pa_draft_revenues_all
pa_draft_invoice_items
pa_draft_revenue_items
pa_agreements_all
pa_bill_groups_all
pa_expenditure_items_all
pa_events
pa_cust_event_rdl_all
pa_cust_rev_dist_lines_all
hr_all_organization_units_tl
pa_expenditure_types_all
pa_event_types
hz_parties
ar_receivable_applications_all
ar_cash_receipts_all
xla_events
xla_ae_headers
```

ECC_SPEC_ID Columns

```
PRJ.PROJECT_ID
QPDR.DRAFT_REVENUE_NUM
CRDL.EXPENDITURE_ITEM_ID
CRDL.LINE_NUM
QPDI.DRAFT_INVOICE_NUM
EV.EVENT_ID
ERDL.LINE_NUM
```

ECC_SPEC_ID Concatenation Structure

```
1.
'PROJ-' || TO_CHAR(PRJ.PROJECT_ID) || '-R-' || TO_CHAR(QPDR.
DRAFT_REVENUE_NUM) || '-EI-' || TO_CHAR(CRDL.EXPENDITURE_ITEM_ID) || '-'
|| TO_CHAR(CRDL.LINE_NUM) || '-I-' || TO_CHAR(NVL(QPDI.
DRAFT_INVOICE_NUM, 0))

2.
'PROJ-' || TO_CHAR(PRJ.PROJECT_ID) || '-R-' || TO_CHAR(QPDR.
DRAFT_REVENUE_NUM) || '-EV-' || TO_CHAR(EV.EVENT_ID) || '-' || TO_CHAR
(ERDL.LINE_NUM) || '-I-' || TO_CHAR(NVL(QPDI.DRAFT_INVOICE_NUM, 0))

3.
'PROJ-' || TO_CHAR(PRJ.PROJECT_ID) || '-I-' || TO_CHAR(QPDI.
DRAFT_INVOICE_NUM) || '-EI-' || TO_CHAR(EI.EXPENDITURE_ITEM_ID) || '-'
|| TO_CHAR(CRDL.LINE_NUM) || '-R-' || TO_CHAR(NVL(QPDR.
DRAFT_REVENUE_NUM, 0))

4.
'PROJ-' || TO_CHAR(PRJ.PROJECT_ID) || '-I-' || TO_CHAR(QPDI.
DRAFT_INVOICE_NUM) || '-EV-' || TO_CHAR(EV.EVENT_ID) || '-' || TO_CHAR
(PDII.LINE_NUM) || '-R-' || TO_CHAR(NVL(QPDR.DRAFT_REVENUE_NUM, 0))
```

Data Set Key: `pa-ds-cost-budget`

Data Set Name

Projects Costing: Budgets

Description (Purpose of the Data Set)

Summarized Budget line data for Project, Budget Task and GL Period combination of the latest Baseline budget version for Projects displayed in the Costing dashboard.

Oracle E-Business Suite Table Names

```
pa_projects_all
pa_project_statuses
pa_project_types_all
pa_budget_versions
pa_budget_types
pa_budget_entry_methods
pa_implementations_all
pa_budget_lines
pa_resource_assignments
pa_tasks
hr_all_organization_units_tl
gl_period_statuses
pa_periods_all
```

ECC_SPEC_ID Columns

```
project_id
task_id
period_name
```

ECC_SPEC_ID Concatenation Structure

```
'CB-'|| TO_CHAR(l_projrec_tbl(projrec).project_id)|| '-' ||TO_CHAR(pt.
TASK_ID) || '-' || TO_CHAR(gps.PERIOD_NAME)
```

Data Set Key: `pa-ds-cost-bgtchart`

Data Set Name

Projects Costing: Budget Chart

Description (Purpose of the Data Set)

Summarized Budget and consumptions data for Projects displayed in Costing dashboard.

Oracle E-Business Suite Table Names

```
PA_ECC_COST_BUDGET
gl_period_statuses
```

ECC_SPEC_ID Columns

```
project_id
task_id
period_name
```

ECC_SPEC_ID Concatenation Structure

```
TO_CHAR(CATEGORY)||TO_CHAR(PROJECT_ID)||TO_CHAR(TASK_ID)||TO_CHAR
(GL_PERIOD_NAME)
```

Property Manager

Data Set Key: `n-payterm`

Data Set Name

Lease Pay Terms Data Set

Description (Purpose of the Data Set)

Provides lease, payment term, options, locations, and assets details; also provides all lease-level balances/amounts.

Oracle E-Business Suite Table Names

```
pn_ecc_pay_term_v
pn_leases_all
pn_lease_details_all
pn_payment_terms_all
pn_system_setup_options
pn_lease_stream_temp
po_vendors vendors
po_vendors
pn_streams_all
pn_stream_lines_all
```

ECC_SPEC_ID Columns

```
pl.lease_id,ppt.payment_term_id,psl.period_start_date,psl.
stream_line_id,location.location_id,ps.regime_code
```

ECC_SPEC_ID Concatenation Structure

```
pl.lease_id
|| ppt.payment_term_id
|| TO_CHAR(psl.period_start_date,'YYYYMM')
|| psl.stream_line_id
|| location.location_id
|| ps.regime_code
```

Data Set Key: pn-expenses

Data Set Name

Lease Expenses

Description (Purpose of the Data Set)

Expenses of lease for the last 13-months period.

Oracle E-Business Suite Table Names

```
 pn_streams_all
pn_stream_lines_all
pn_leases_all_v
pn_payment_terms_all_v
pn_lease_details_all_v
pn_eqp_assets_all
pn_system_setup_options
```

ECC_SPEC_ID Columns

```
org_id,lease_id,payment_term_id,stream_type_code,regime_code,
stream_date,period_start_date
```

ECC_SPEC_ID Concatenation Structure

```
org_id
|| lease_id
|| payment_term_id
|| stream_type_code
|| regime_code
|| stream_date
|| TO_CHAR(period_start_date,'YYYYMMDD')
```

Data Set Key: `pn-projection`

Data Set Name

Projection Data Set

Description (Purpose of the Data Set)

Future projection of streams data for lease analysis.

Oracle E-Business Suite Table Names

```
pn_streams_all
pn_stream_lines_all
pn_payment_terms_all_v
pn_leases_all_v
```

ECC_SPEC_ID Columns

```
psa.lease_id,psa.payment_term_id,psa.org_id,psa.regime_code,psl.
stream_date,psa.stream_type_code
```

ECC_SPEC_ID Concatenation Structure

```
psa.lease_id
|| psa.payment_term_id
|| psa.org_id
|| psa.regime_code
|| TO_CHAR(psl.stream_date,'DDMMYYYY')
|| psa.stream_type_code
```

Data Set Key: `pn-normalize`

Data Set Name

Normalized Data for Property Manager Leases

Description (Purpose of the Data Set)

Normalized cash items of payment terms.

Oracle E-Business Suite Table Names

```
pn_payment_items_all
pn_payment_schedules_all
pn_payment_terms_all
pn_eqp_payment_items_all
pn_eqp_payment_schedules_all
pn_eqp_payment_terms_all
```

ECC_SPEC_ID Columns

```
lease_id,schedule_date,currency_code
```

ECC_SPEC_ID Concatenation Structure

```
lease_id
|| TO_CHAR(schedule_date,'YYYYMMDD')
|| currency_code
```

Receivables

Data Set Key: `ar-trx`

Data Set Name

Outstanding Receivables

Description (Purpose of the Data Set)

Complete and unpaid transactions (all transaction types)

Oracle E-Business Suite Table Names

```
ar_payment_schedules_all
```

ECC_SPEC_ID Columns

```
payment_schedule_id
```

ECC_SPEC_ID Concatenation Structure

```
'TRX-' || payment_schedule_id
```

Data Set Key: `ar-billproc`

Data Set Name

Billing Process

Description (Purpose of the Data Set)

Incomplete transactions (all transaction types), adjustments pending approval, and import errors

Oracle E-Business Suite Table Names

```
ar_adjustments_all adjust
ra_customer_trx_all trx
ar_transmissions_all transms
```

ECC_SPEC_ID Columns

```
adjust.adjustment_id
trx.customer_trx_id
transms.transmission_id
```

ECC_SPEC_ID Concatenation Structure

```
'ARADJUST-' || adjust.adjustment_id
'ARIMCPTRX-' || trx.customer_trx_id
'ARLOCKBOX-' || transms.transmission_id
```

Data Set Key: `ar-pmtproc`

Data Set Name

Payment Process

Description (Purpose of the Data Set)

Unapplied receipts (all receipt types)

Oracle E-Business Suite Table Names

```
ar_batches_all batches
ar_cash_receipts_al receipt
ar_batches_all batches
```

ECC_SPEC_ID Columns

```
batches.batch_id
receipt.cash_receipt_id
batches.batch_id
```

ECC_SPEC_ID Concatenation Structure

```
'ARRCPTBATCH-' || batches.batch_id
'ARCASHRCPT-' || receipt.cash_receipt_id
'ARRMITBATCH-' || batches.batch_id
```

Data Set Key: `ar-history`

Data Set Name

Payment History

Description (Purpose of the Data Set)

Fully paid transactions and fully applied receipts

Oracle E-Business Suite Table Names

```
ar_cash_receipts_al receipt
ar_payment_schedules_all ps
```

ECC_SPEC_ID Columns

```
receipt.cash_receipt_id
ps.payment_schedule_id
```

ECC_SPEC_ID Concatenation Structure

```
'ARRCIPTHSTRY-' || receipt.cash_receipt_id
'ARTRXHSTRY-' || ps.payment_schedule_id
```

Data Set Key: `ar-dispute`

Data Set Name

Disputes

Description (Purpose of the Data Set)

Customer disputes (all dispute statuses)

Oracle E-Business Suite Table Names

```
ra_cm_requests_all cmreq
```

ECC_SPEC_ID Columns

```
cmreq.request_id
```

ECC_SPEC_ID Concatenation Structure

```
'CMREQ' || cmreq.request_id
```

Data Set Key: `ar-period-close`

Data Set Name

AR Closing

Description (Purpose of the Data Set)

Unaccounted transactions and receipts.

Pending accounting events for transactions and receipts (draft and invalid accounting)
Accounted transactions and receipts but not transferred yet to GL.

Oracle E-Business Suite Table - Unaccounted Transactions

ECC_SPEC_ID Columns

```
rct.customer_trx_id
arp.PAYMENT_SCHEDULE_ID
xe.event_id
xe.event_number
```

ECC_SPEC_ID Concatenation Structure

```
rct.customer_trx_id||'-'||arp.PAYMENT_SCHEDULE_ID||'-'||xe.
event_id||'-'||xe.event_number
```

Oracle E-Business Suite Table - Accounted not transferred to GL Transaction

ECC_SPEC_ID Columns

```
rct.customer_trx_id
xe.event_id
xe.event_number
xah.ae_header_id
xal.ae_line_num
```

ECC_SPEC_ID Concatenation Structure

```
rct.customer_trx_id||'-'||xe.event_id||'-'||xe.event_number||'-'||xah.
ae_header_id||'-'||xal.ae_line_num
```

Oracle E-Business Suite Table - Unaccounted Receipts

ECC_SPEC_ID Columns

```
acra.cash_receipt_id
acrha.cash_receipt_history_id
xe.event_id
xe.event_number
```

ECC_SPEC_ID Concatenation Structure

```
acra.cash_receipt_id||'-'||acrha.cash_receipt_history_id||'-'||xe.
event_id||'-'||xe.event_number
```

Oracle E-Business Suite Table - Accounted not transferred to GL Applied Receipts

ECC_SPEC_ID Columns

```
acra.cash_receipt_id
acrha.cash_receipt_history_id
xe.event_id
xe.event_number
xah.ae_header_id
xal.ae_line_num
```

ECC_SPEC_ID Concatenation Structure

```
acra.cash_receipt_id||'-'||acrha.cash_receipt_history_id||'-'||xe.
event_id||'-'||xe.event_number||'-'||xah.ae_header_id||'-'||xal.
ae_line_num
```

Oracle E-Business Suite Table - Accounted not transferred to GL Unapplied and Unidentified Receipts

ECC_SPEC_ID Columns

```
acra.cash_receipt_id
acrha.cash_receipt_history_id
xe.event_id
xe.event_number
xah.ae_header_id
xal.ae_line_num
```

ECC_SPEC_ID Concatenation Structure

```
acra.cash_receipt_id||'-'||acrha.cash_receipt_history_id||'-'||xe.
event_id||'-'||xe.event_number||'-'||xah.ae_header_id||'-'||xal.
ae_line_num
```

Service

Data Set Key: `cs-service`

Data Set Name

Service Request

Description (Purpose of the Data Set)

Service Request and associate contact information.

Oracle E-Business Suite Table Names

```
cs_incidents_all_b,cs_incidents_all_tl,
cs_incident_types_tl,cs_incident_statuses_tl,
jtf_rs_resource_extns,jtf_rs_groups_b,
jtf_rs_groups_tl,cs_incident_urgencies_tl,
cs_incident_severities_tl, hz_parties,
hz_cust_accounts, fnd_territories_tl ,
cs_hz_sr_contact_points,mtl_system_items_b_kfv,
mtl_system_items_tl, mtl_categories_kfv ,
csi_item_instances, csi_systems_tl,
okc_k_headers_all_b, okc_statuses_tl,
okc_k_lines_b, okc_k_lines_tl,
oks_cov_types_b, oks_cov_types_tl,
jtf_tasks_b, jtf_task_references_b,
fnd_lookup_values, jtf_task_statuses_b,
jtf_task_statuses_tl, hz_relationships,
per_all_people_f, per_phones,
hz_locations, hz_party_sites,
jtf_rs_resource_extns_tl
```

ECC_SPEC_ID Columns

```
incident_id
```

ECC_SPEC_ID Concatenation Structure

```
incident_id
```

Data Set Key: `cs-task`

Data Set Name

Service Task

Description (Purpose of the Data Set)

Task details for service requests.

Oracle E-Business Suite Table Names

```
cs_incidents_all_b,hz_parties,
jtf_tasks_b,jtf_tasks_tl,
jtf_task_types_tl,jtf_task_types_b,
jtf_task_statuses_tl,jtf_task_statuses_b,
jtf_task_priorities_tl,jtf_objects_tl,
hz_locations,hz_party_sites,
fnd_lookup_values,jtf_task_assignments,
jtf_task_all_assignments,jtf_rs_teams_tl,
jtf_rs_resource_extns_tl,jtf_rs_groups_tl,
jtf_rs_resource_extns,jtf_rs_salesreps
```

ECC_SPEC_ID Columns

```
task_assignment_id
```

ECC_SPEC_ID Concatenation Structure

```
task_assignment_id
```

Data Set Key: `cs-debrief`

Data Set Name

Service Debrief

Description (Purpose of the Data Set)

Service Debrief.

Oracle E-Business Suite Table Names

```
csf_debrief_lines ,
csf_debrief_headers  ,
jtf_task_all_assignments ,
jtf_tasks_b ,
cs_incidents_all_b ,
mtl_system_items_b_kfv ,
cs_billing_type_categories ,
cs_transaction_types_b ,
cs_transaction_types_tl ,
mtl_units_of_measure_tl
```

ECC_SPEC_ID Columns

```
debrief_header_id,
debrief_line_id
```

ECC_SPEC_ID Concatenation Structure

```
debrief_header_id||'-'||cdl.debrief_line_id
```

Data Set Key: `cs-srnote`

Data Set Name

Service Notes

Description (Purpose of the Data Set)

Service Notes.

Oracle E-Business Suite Table Names

```
jtf_notes_b ,
jtf_notes_tl ,
cs_incidents_all_b
```

ECC_SPEC_ID Columns

```
jtf_note_id
```

ECC_SPEC_ID Concatenation Structure

```
jtf_note_id
```

Data Set Key: `cs-mcontact`

Data Set Name

Service Requests contacts

Description (Purpose of the Data Set)

Service Requests contacts.

Oracle E-Business Suite Table Names

```
cs_hz_sr_contact_points ,
cs_incidents_all_b ,
cs_incidents_all_tl
```

ECC_SPEC_ID Columns

```
sr_contact_point_id
```

ECC_SPEC_ID Concatenation Structure

```
sr_contact_point_id
```

Data Set Key: `cs-mproduct`

Data Set Name

Service Requests Products

Description (Purpose of the Data Set)

Service Requests Products.

Oracle E-Business Suite Table Names

```
cs_sr_products,
mtl_system_items_b_kfv,
mtl_system_items_tl ,
csi_item_instances ,
cs_incidents_all_b ,
cs_incidents_all_tl ,
mtl_categories_kfv ,
okc_k_headers_all_b ,
okc_statuses_tl ,
okc_k_lines_b ,
okc_k_lines_tl ,
oks_k_lines_b ,
oks_k_lines_b ,
oks_cov_types_b ,
oks_cov_types_tl
```

ECC_SPEC_ID Columns

```
sr_product_id
```

ECC_SPEC_ID Concatenation Structure

```
sr_product_id
```

Data Set Key: `cs-charges`

Data Set Name

Service Charges

Description (Purpose of the Data Set)

Charged Details for Service Requests.

Oracle E-Business Suite Table Names

```
cs_estimate_details ced
cs_transaction_types_tl seractivities
cs_business_processes busprocess
cs_incidents_all_b inc
cs_txn_billing_types billingtype
cs_incidents_all_tl tl
hz_parties hz
hz_cust_accounts acc
hr_all_organization_units_tl hr
qp_list_headers_tl qp
okc_k_headers_all_b hdr
okc_k_lines_b corelines
oe_order_headers_all oe
oe_order_lines_all oel
cs_cost_details ccd
csi_item_instances cp
mtl_system_items_b_kfv items
csi_item_instances csi2
```

ECC_SPEC_ID Columns

```
estimate_detail_id
```

ECC_SPEC_ID Concatenation Structure

```
estimate_detail_id
```

Service Contracts (OKS)

Data Set Key: `oks-contract`

Data Set Name

Service Contract Details

Description (Purpose of the Data Set)

Provides information for the contract header, lines, and sublines.

Oracle E-Business Suite Table Names

```
oks_k_ecc_header_v hdr
oks_k_ecc_line_v line
oks_k_ecc_subline_v sline
```

ECC_SPEC_ID Columns

```
hdr.contract_operating_unit_id,
hdr.contract_id,
line.line_id,
sline.sub_line_id
```

ECC_SPEC_ID Concatenation Structure

```
to_char (hdr.contract_operating_unit_id)
|| '-'
|| to_char (hdr.contract_id)
|| '-'
|| to_char (line.line_id)
|| '-'
|| to_char (sline.sub_line_id)
```

Data Set Key: `oks-contract-sr`

Data Set Name

Service Contract Service Request Details

Description (Purpose of the Data Set)

Provides details of the service requests.

Oracle E-Business Suite Table Names

```
cs_incident_types_tl type_tl
cs_incident_types_b type
cs_incident_severities_tl sev_tl
cs_incident_severities_b sev
cs_incident_statuses_tl status_tl
cs_incident_statuses_b status
cs_incident_urgencies_b urgency
cs_incidents_all_vl inc
hz_parties hzp
okc_k_headers_all_b oks_hdr
okc_statuses_b sts
```

ECC_SPEC_ID Columns

```
inc.incident_id,
inc.incident_type_id,
inc.org_id
```

ECC_SPEC_ID Concatenation Structure

```
'SR - '
|| to_char (inc.incident_id)
|| ' - '
|| to_char (inc.incident_type_id)
|| ' - '
|| to_char (inc.org_id)
```

Data Set Key: `oks-contract-contact`

Data Set Name

Service Contract Contact Details

Description (Purpose of the Data Set)

Provides information about the contact details provided in the service contracts.

Oracle E-Business Suite Table Names

```
okc_contacts oc_sale
okx_party_contacts_v opc
fnd_lookup_values fndv
okc_k_headers_all_b okh
okc_statuses_b sts
```

ECC_SPEC_ID Columns

```
dnz_chr_id
```

ECC_SPEC_ID Concatenation Structure

```
dnz_chr_id
```

Data Set Key: `oks-contract-renewal`

Data Set Name

Service Contract Renewal Details

Description (Purpose of the Data Set)

Fetches the data of the renewed contracts to calculate On Time Renewal and Renewal Rate.

Oracle E-Business Suite Table Names

```
okc_k_headers_all_b okchdr
oks_k_headers_b khdr
okc_statuses_b okcsts
okc_operation_lines oplines
okc_operation_instances opinst
okc_class_operations classops
```

ECC_SPEC_ID Columns

```
okchdr.id
```

ECC_SPEC_ID Concatenation Structure

```
okchdr.id
```

Data Set Key: `oks-contract-salescredit`

Data Set Name

Service Contract Sales Credit Details

Description (Purpose of the Data Set)

Provides the details of the Sales Credits.

Oracle E-Business Suite Table Names

```
oks_k_sales_credits linesales
jtf_rs_salesreps linejtfrs
jtf_rs_groups_tl linegrp
oe_sales_credit_types lineoe
okc_k_headers_all_b salelinehdr
okc_k_lines_b saleline
```

ECC_SPEC_ID Columns

```
linesales.id
```

ECC_SPEC_ID Concatenation Structure

```
linesales.id
```

Data Set Key: `oks-contract-customer`

Data Set Name

Service Contract Customer Details

Description (Purpose of the Data Set)

Provides more information about the customer entered in the service contract.

Oracle E-Business Suite Table Names

```
HZ_ORGANIZATION_PROFILES Orgpro ,
HZ_PARTIES Party ,
HZ_CREDIT_RATINGS Credit,
HZ_PARTY_SITES ps,
HZ_LOCATIONS loc,
oks_endeca_tca_finrep_v tca_finrep ,
okc_k_party_roles_b CUSTOMER,
okc_k_headers_all_b c
```

ECC_SPEC_ID Columns

```
c.id
```

ECC_SPEC_ID Concatenation Structure

```
c.id
```

Data Set Key: `oks-counter`

Data Set Name

Service Contract Counter Details

Description (Purpose of the Data Set)

Provides information related to the counters such as Counter Groups, Latest Reading, and so on, covered under the Usage lines.

Oracle E-Business Suite Table Names

```
okc_k_headers_all_b hdr,
okc_k_lines_b okcline,
okc_k_lines_b okcsline,
oks_k_lines_b okssline,
okc_k_items  oktsline,
CSI_COUNTERS_B           CT,
csi_counters_tl cct,
CS_CSI_COUNTER_GROUPS    CG,
CSI_COUNTER_ASSOCIATIONS CCA,
mtl_system_items_kfv item
```

ECC_SPEC_ID Columns

```
hdr.contract_operating_unit_id,
hdr.contract_id,
line.line_id,
sline.sub_line_id
```

ECC_SPEC_ID Concatenation Structure

```
to_char (hdr.org_id)
        || '-'
        || to_char (hdr.id)
        || '-'
        || to_char (okcline.id)
        || '-'
        || to_char (okcsline.id)
```

Data Set Key: `oks-followup`

Data Set Name

Service Contracts Follow-Up Details

Description (Purpose of the Data Set)

This dataset provides information about the Follow-Up details.

Oracle E-Business Suite Table Names

```
OKS_FOLLOW_UP_ACTIONS
```

ECC_SPEC_ID Columns

```
id
```

ECC_SPEC_ID Concatenation Structure

```
id
```

Data Set Key: `oks-bill-details`

Data Set Name

Service Contract Billing Details

Description (Purpose of the Data Set)

This dataset provides information about the usage billing.

Oracle E-Business Suite Table Names

```
oks_bill_sub_lines  okbs,
okc_k_lines_b okcl
```

ECC_SPEC_ID Columns

```
okbs.id
```

ECC_SPEC_ID Concatenation Structure

```
okbs.id
```

Data Set Key: `oks-contract-ren-pred`

Data Set Name

Service Contracts Renewal Prediction

Description (Purpose of the Data Set)

This dataset provides information about the attributes influencing the probability score of the Renewal Prediction.

Oracle E-Business Suite Table Names

```
oks_ml_renew_pred_attr okbs,
okl_ml_attributes_b okla,
okl_ml_attributes_tl okltl,
oks_ml_renew_pred_results mlpred,
okc_k_lines_b okl
```

ECC_SPEC_ID Columns

```
okbs.contract_line_id
attr_name
```

ECC_SPEC_ID Concatenation Structure

```
okbs.contract_line_id||attr_name
```

Sourcing

Data Set Key: `pon-pcc-neg`

Data Set Name

Negotiations

Description (Purpose of the Data Set)

This data set provides information on negotiation and their details.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all
pon_neg_team_members
pon_auction_item_prices_all
pon_bid_headers
pon_bid_item_prices
```

ECC_SPEC_ID Columns

```
AH.AUCTION_HEADER_ID
NEG_TEAM.USER_ID
PITMDS.ITEM_LINE_NUMBER
PBIDS.BID_NUMBER
PBIDS.BIDLINE_NUMBER
```

ECC_SPEC_ID Concatenation Structure

```
AH.AUCTION_HEADER_ID||'-'||NEG_TEAM.USER_ID||'-'||PITMDS.
ITEM_LINE_NUMBER||'-'||PBIDS.BID_NUMBER||'-'||PBIDS.BIDLINE_NUMBER
```

Data Set Key: `pon-pcc-neg-time`

Data Set Name

Negotiation Timeline

Description (Purpose of the Data Set)

This data set provides information on time line for each stage of negotiation process.

Oracle E-Business Suite Table Names

```
pon_auction_headers_all
```

ECC_SPEC_ID Columns

```
auction_header_id
```

ECC_SPEC_ID Concatenation Structure

```
auction_header_id||'-'||duration_code
```

Index