Oracle Human Capital Management Cloud
Integrating with HCM

19C
# Contents

## Preface

| Preface | i |

## 1 Introduction to Integrating with Oracle HCM Cloud

| Overview of Integrating with Oracle HCM Cloud | 1 |
| Inbound Integrations | 5 |
| Outbound Integrations | 7 |

## 2 Introduction to HCM Data Loader

| Overview of HCM Data Loader | 9 |
| HCM Data Loader and Implementation Scenarios | 10 |
| How Data Is Loaded Using HCM Data Loader | 11 |
| Supported Key Types | 12 |
| Source Keys | 13 |
| User Keys | 14 |
| Oracle Fusion GUIDs and Surrogate IDs | 15 |
| Business-Object Structure | 15 |
| FAQs for Introduction to HCM Data Loader | 17 |

## 3 Data File Instructions and Delivery

| HCM Data Loader Data and .zip Files | 19 |
| File Line Instructions and File Discriminators | 20 |
| The METADATA Instruction | 22 |
| The MERGE Instruction | 23 |
| The DELETE Instruction | 24 |
| The SET Instruction | 24 |
| Generate and Use Template Files | 27 |
| FAQs for Data File Instructions and Delivery | 29 |
# Preparing to Use HCM Data Loader

- Sources of Business-Object Information
- HCM Data Loader Configuration Parameters
- HCM Data Loader Scope
- How You Define Referenced Objects
- How You Review Lists of Values
- Define the Source-System Owner
- Guidelines for Preparing to Load Workers
- How You Test the HCM Data Loader Process Flow and Connections
- How You Audit Objects Loaded in Bulk
- FAQs for Preparing to Use HCM Data Loader

# Data Preparation

- Guidelines for Preparing the Source Data
- General Data Considerations for HCM Data Loader
- How You Load Images, Attachments, and Large Strings
- How You Supply Source Keys
- How You Update Source Keys
- How You Supply User Keys
- Examples of Supplying Oracle Fusion Surrogate IDs
- Examples of Supplying Oracle Fusion GUIDs
- How You Manage Reserved Characters
- How You Load and Maintain Translated Objects
- How You Include Source-System References in Data Files
- Data Deletion Using HCM Data Loader

# Loading Date-Effective Data

- Overview of Loading Date-Effective Objects
- How You Load Objects with Multiple Changes per Day
- Options for Updating Date-Effective Objects with Future-Dated Records
- Example of Retaining Attribute Values in Future-Dated Records
- Options for Carrying Forward Changed Attribute Values in Retain Mode
- Example of Replacing Attribute Values in Future-Dated Records
- Examples of Changing First Effective Start Date and Last Effective End Date
7 Loading Flexfield Data

Guidelines for Loading Flexfield Data 71
How You Load Data for Descriptive Flexfields 72
Example of Loading Data for Descriptive Flexfields 73
How You Load Data for Extensible Flexfields 74
Example of Loading Data for Extensible Flexfields 75

8 Loading Setup Objects

Guidelines for Loading Absence Cases 77
Examples of Loading Actions 77
Examples of Loading Action Reasons 78
Guidelines for Loading Calendar Events 79
Guidelines for Loading Checklist Templates 80
Examples of Loading Checklist Templates 81
Guidelines for Loading Document Delivery Preferences 83
Guidelines for Loading Document Types 84
Examples of Loading Document Types 85
Examples of Loading Extended Lookup Codes 87
Guidelines for Loading Name Formats 88
Examples of Loading Person Types 90
Examples of Loading Resource Exceptions 91
Guidelines for Loading Time and Labor Event Groups 92
Examples of Loading Time and Labor Event Groups 93
Columns and Event Source Objects for Time and Labor Events 94
Examples of Loading Time and Labor Event Actions 96
## Loading Workers

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview of Loading Workers</td>
<td>99</td>
</tr>
<tr>
<td>Guidelines for Loading New Workers</td>
<td>101</td>
</tr>
<tr>
<td>Effective Start and End Dates for Worker Components</td>
<td>102</td>
</tr>
<tr>
<td>Guidelines for Deleting Worker Components</td>
<td>103</td>
</tr>
<tr>
<td>How You Check for Duplicate Person Records</td>
<td>104</td>
</tr>
<tr>
<td>Person Numbers in Worker Objects</td>
<td>105</td>
</tr>
<tr>
<td>Guidelines for Loading External Identifiers</td>
<td>106</td>
</tr>
<tr>
<td>Guidelines for Loading Person Names</td>
<td>107</td>
</tr>
<tr>
<td>Guidelines for Loading Person Images</td>
<td>108</td>
</tr>
<tr>
<td>Examples of Loading Work Relationship Changes</td>
<td>108</td>
</tr>
<tr>
<td>How You Perform a Global Transfer</td>
<td>110</td>
</tr>
<tr>
<td>Guidelines for Loading Additional Assignments</td>
<td>111</td>
</tr>
<tr>
<td>Guidelines for Loading Temporary Assignments</td>
<td>112</td>
</tr>
<tr>
<td>Start Dates for Person Legislative Data</td>
<td>113</td>
</tr>
<tr>
<td>Employment Terms Override at Assignment</td>
<td>114</td>
</tr>
<tr>
<td>Automatic Calculation of FTE Values for Workers</td>
<td>114</td>
</tr>
<tr>
<td>Create a Default Working-Hour Pattern</td>
<td>115</td>
</tr>
<tr>
<td>Examples of Loading Worker Working Hour Pattern</td>
<td>115</td>
</tr>
<tr>
<td>Examples of Loading Assignment Eligible Jobs</td>
<td>116</td>
</tr>
<tr>
<td>Guidelines for Loading Seniority Dates</td>
<td>117</td>
</tr>
<tr>
<td>Examples of Loading Seniority Date Adjustments</td>
<td>119</td>
</tr>
<tr>
<td>Examples of Loading Seniority Hours</td>
<td>121</td>
</tr>
<tr>
<td>Example of Loading a Contract Extension</td>
<td>122</td>
</tr>
<tr>
<td>Example of Loading a Default Expense Account for a Worker</td>
<td>123</td>
</tr>
<tr>
<td>FAQs for Loading Workers</td>
<td>124</td>
</tr>
</tbody>
</table>
10  Loading Worker-Related Objects

Example of Loading an Allocated Checklist
Guidelines for Loading Areas of Responsibility
Guidelines for Loading Document Records
Guidelines for Loading Time Record Groups
Examples of Loading Time Record Groups
Guidelines for Loading User-Update Requests
Examples of Loading User-Update Requests
Guidelines for Loading Worker Schedules
Examples of Creating Worker Schedules
Examples of Updating Worker Schedules
FAQs for Loading Worker-Related Objects

11  Loading Work Structures

Overview of Loading Work Structures
Examples of Loading Collective Agreements
Guidelines for Loading Grades
Guidelines for Loading Grade Ladders
Guidelines for Loading Grade Rates
Guidelines for Loading Jobs
Examples of Loading Job Families
Guidelines for Loading Locations
Guidelines for Loading Organizations
Guidelines for Loading Positions
Examples of Loading HCM Position Hierarchies
Guidelines for Loading Department Trees
Guidelines for Loading Department Tree Nodes
Examples of Loading Department Tree Nodes
Guidelines for Loading Organization Trees
Guidelines for Loading Organization Tree Nodes
Examples of Loading Organization Tree Nodes
### Loading Compensation Objects

Guidelines for Loading Market Data Objects

Examples of Loading Market Data Career Level, Career Stream, and Other Level

Examples of Loading Market Data Job Family and Function

Examples of Loading Market Data Location List and Job List

Example of Loading Market Data Survey Data

Guidelines for Loading Progression Grade Ladders

Examples of Loading Progression Grade Ladders with Steps

Examples of Loading Progression Grade Ladders Without Steps

Guidelines for Loading Progression Grade Rates

Examples of Loading Progression Grade Rates

Guidelines for Loading Progression Rules

Examples of Loading Progression Rules

Guidelines for Loading Salary Basis Records

Guidelines for Loading Salary Records

Examples of Loading Salary Records

Examples of Loading Salary Range Differential Profiles and Values

Guidelines for Making Date-Effective Changes to Salary Range Differentials

---

### Loading Payroll Relationships

Payroll Information

Assigned Payroll

Payroll Assignments

---

### Loading Payroll Setup Objects

Examples of Loading Legislative Data Groups

Object Groups

Example of Loading Payroll Consolidation Group

Examples of Loading Payroll Definitions and Time Periods

Example of Loading Time Definitions

Example of Loading Payroll Element Run Usage

User-Defined Tables

Fast Formulas

Elements

Element Entries
### 15 Loading Balance Definitions

- Examples of Loading Payroll Balance Definitions 229
- Payroll Balance Groups 231
- Balance Initializations 237
- Balance Adjustments 243

### 16 Loading Payment Methods

- Examples of Loading Banks 251
- Examples of Loading Bank Branches 251
- Examples of Loading External Bank Accounts 252
- Example of Loading Organization Payment Methods 253
- Example of Loading Personal Payment Methods 255
- Example of Loading Third-Party Organization Payment Method 257
- Example of Loading Third-Party Personal Payment Method 258

### 17 Loading Payroll Costing

- Overview of Loading Payroll Costing 261
- Example of Loading Element Eligibility Costing 263
- Example of Loading Costing of Payroll 266
- Example of Loading Costing of Departments 267
- Example of Loading Costing of Jobs 270
- Example of Loading Costing of Positions 271
- Example of Loading Payment Source Costing 271
- Example of Loading Person Costing 272
- Example of Loading Person Costing at Payroll Relationship Level 273
- Example of Loading Person Costing at Assignment Level 273
- Example of Loading Person Element Costing at Assignment Level 274
- Example of Loading Person Element Costing at Payroll Relationship Level 275
- Example of Loading Element Entry Costing 275

### 18 Using Transformation Formulas

- HCM Data Loader Transformation Formula 277
- HCM Spreadsheet Data Loader 289
22 Loading Benefits Objects

Example of Loading Beneficiary Enrollments 323
Example of Loading Dependent Enrollments 324
Example of Loading Participant Enrollments 325
Example of Loading Person Benefit Group 326
Example of Loading Person Benefit Balance 327
Example of Loading Person Habits 327

23 Importing and Loading Data Using HCM Data Loader

How Data Is Imported and Loaded 329
Import and Load Data 330
Upload Data Using HCM Data Loader 332
How You Review and Correct Errors 334
Stop HCM Data Loader Processing 336
Overview of Automating Data Loading 337
FAQs for Importing and Loading Data Using HCM Data Loader 338

24 Data Loading Using Spreadsheets

Spreadsheet Data Loaders 341
Set Up Desktop Integration for Excel 342
Troubleshoot Desktop Integration for Excel 343

25 Introduction to HCM Spreadsheet Data Loader

Guidelines for Using HCM Spreadsheet Data Loader 345
How Data Is Uploaded Using HCM Spreadsheet Data Loader 346
How You Test the HCM Spreadsheet Data Loader Process Flow and Connections 348
Spreadsheet Business Object Access 349
Manage Spreadsheet Business Object Access 350
How You Enable Access to HCM Spreadsheet Data Loader Tasks 351
FAQs for HCM Spreadsheet Data Loader 353
30 HCM Extracts and Diagnostics for HCM Data Loader and HCM Spreadsheet Data Loader

HCM Data Loader Extracts 395
Setting Up HCM Data Loader Extracts 395
Running HCM Data Loader Extracts 400
Diagnostic Tests for HCM Data Loader 402
Diagnostic Tests for HCM Spreadsheet Data Loader 404

31 Best Practices for HCM Data Loader and HCM Spreadsheet Data Loader 407

HCM Data Loader Best Practices 407
Overview of Maintaining the Stage Tables 408
Example of the Automatic Deletion and Backup of Stage Table Data 410
Maintain the HCM Data Loader Stage Tables 412
Maintain the HCM Spreadsheet Data Loader Stage Tables 414

32 Time, Absence, and Pension Data for Payroll 417

Overview 417
Prerequisite Payroll Setup for Importing Time Entries: Explained 418
Create Elements for Time Card Entries 420
Importing Time Card Entries to Payroll: Procedure 423
File Format for Importing Time Entries 425
Integrating Absence Management with Global Payroll: Procedure 427
Define Payroll Elements to Process Absences 429
Importing Absence Entries to Payroll: Procedure 432
File Format for Importing Absence Entries 434
Running the Load Benefit Batches Process: Procedure 436
File Format for Importing Pension Deductions 438
36 Benefits Data Extract

Guidelines for Extracting Benefits Data Using HCM Extracts 487
Configure Benefits for Extracting Data 487
Valid Extract Names 490
Generate and Transmit Benefit Extracts 491
Example of a Custom Layout for Benefits Extract 492

37 Talent Review and Succession Management Data Extract

Guidelines for Extracting Talent Review Data Using HCM Extracts 499
Guidelines for Extracting Succession Management Data Using HCM Extracts 500

38 Transferring Encrypted Data

Set up Encryption for File Transfer 503
Encrypt and Upload Files Automatically 505
Transfer Files Automatically from HCM Extracts with PGP Encryption 506

39 Oracle Taleo Recruiting Cloud Service Integration

HCM and Oracle Taleo Recruiting Cloud Service Components: How They Work Together 509
Importing Data From Oracle Taleo Recruiting Cloud Service: Explained 509
Exporting Data to Oracle Taleo Recruiting Cloud Service: Explained 510
FAQs for Oracle Taleo Recruiting Cloud Service and HCM Integration 512

40 Web Services

Overview of Web Services 513
Preface

This preface introduces information sources that can help you use the application.

Using Oracle Applications

Using Applications Help

Use help icons  to access help in the application. If you don’t see any help icons on your page, click your user image or name in the global header and select Show Help Icons. Not all pages have help icons. You can also access Oracle Applications Help.

Watch: This video tutorial shows you how to find help and use help features.

You can also read Using Applications Help.

Additional Resources

• Community: Use Oracle Cloud Customer Connect to get information from experts at Oracle, the partner community, and other users.

• Guides and Videos: Go to the Oracle Help Center to find guides and videos.

• Training: Take courses on Oracle Cloud from Oracle University.

Conventions

The following table explains the text conventions used in this guide.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>boldface</td>
<td>Boldface type indicates user interface elements, navigation paths, or values you enter or select.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates file, folder, and directory names, code examples, commands, and URLs.</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than symbol separates elements in a navigation path.</td>
</tr>
</tbody>
</table>

Documentation Accessibility

For information about Oracle’s commitment to accessibility, visit the Oracle Accessibility Program website.

Videos included in this guide are provided as a media alternative for text-based help topics also available in this guide.
Contacting Oracle

Access to Oracle Support
Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit My Oracle Support or visit Accessible Oracle Support if you are hearing impaired.

Comments and Suggestions
Please give us feedback about Oracle Applications Help and guides! You can send an e-mail to: oracle_fusion_applications_help_ww_grp@oracle.com.
1 Introduction to Integrating with Oracle HCM Cloud

Overview of Integrating with Oracle HCM Cloud

This guide describes the ways in which you can integrate applications with Oracle HCM Cloud. Most of the content describes ways of loading or extracting data.

This topic shows the scope of this guide and summarizes the contents of each chapter.

Introduction

This table describes Chapter 1.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction to Integrating with Oracle HCM Cloud</td>
<td>An introduction to this guide and an overview of the available approaches to data loading and extracts.</td>
</tr>
</tbody>
</table>

Data Loading Using HCM Data Loader

This table describes chapters related to bulk data loading with HCM Data Loader.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Introduction to HCM Data Loader</td>
<td>An introduction to HCM Data Loader and the data-loading process.</td>
</tr>
<tr>
<td>3</td>
<td>Data File Instructions and Delivery</td>
<td>Valid instructions to include in data files, how to deliver data files for upload, and how to generate and use business-object templates.</td>
</tr>
<tr>
<td>4</td>
<td>Preparing to Use HCM Data Loader</td>
<td>Setting configuration parameters, defining referenced objects, reviewing lists of values, and finding business-object information.</td>
</tr>
<tr>
<td>5</td>
<td>Data Preparation</td>
<td>General guidance for preparing various types of data, such as images and attachments, to ensure successful loading. Information about specifying the supported key types.</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
<td>Contents</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>6</td>
<td>Loading Date-Effective Data</td>
<td>How to load date-effective objects successfully and achieve the required results.</td>
</tr>
<tr>
<td>7</td>
<td>Loading Flexfield Data</td>
<td>How to load data for descriptive and extensible flexfields.</td>
</tr>
<tr>
<td>8</td>
<td>Loading Setup Objects</td>
<td>How to load setup objects, such as Actions, Action Reasons, Name Format, and Person Type.</td>
</tr>
<tr>
<td>9</td>
<td>Loading Workers</td>
<td>How to load components of the Worker object.</td>
</tr>
<tr>
<td>10</td>
<td>Loading Worker-Related Objects</td>
<td>How to load worker-related objects, such as Area of Responsibility, Document Record, and User.</td>
</tr>
<tr>
<td>11</td>
<td>Loading Work Structures</td>
<td>How to load work structures, such as Grade, Job, and Organization.</td>
</tr>
<tr>
<td>12</td>
<td>Loading Compensation Objects</td>
<td>How to load compensation objects, such as Progression Grade Ladder, Salary Basis, and Salary.</td>
</tr>
<tr>
<td>13</td>
<td>Loading Payroll Relationships</td>
<td>How to load payroll information at the payroll relationship, payroll assignment records, and at assigned payroll levels.</td>
</tr>
<tr>
<td>14</td>
<td>Loading Payroll Setup Objects</td>
<td>How to load payroll setup objects, such as object groups, payroll consolidation groups, payroll definitions, time definitions, user-defined tables, fast formula, and elements.</td>
</tr>
<tr>
<td>15</td>
<td>Loading Balance Definitions</td>
<td>How to load payroll balance groups and to initialize, and adjust balances.</td>
</tr>
<tr>
<td>16</td>
<td>Load Payment Methods</td>
<td>How to load banks, bank branches, external bank accounts, and organization and personal payment methods.</td>
</tr>
<tr>
<td>17</td>
<td>Payroll Costing</td>
<td>How to load payroll costing setup details for different payroll accounts.</td>
</tr>
<tr>
<td>18</td>
<td>Using Transformation Formula</td>
<td>How to use the HCM Data Loader Transformation formula to transform your data into a format that is supported by HCM Data Loader. Also, how to use the HCM Spreadsheet Data Loader to load all payroll objects that HCM Data Loader supports.</td>
</tr>
<tr>
<td>19</td>
<td>Loading Talent Objects</td>
<td>How to load talent objects, such as Goal and Goal Plan.</td>
</tr>
</tbody>
</table>
## Data Loading Using Spreadsheet Loaders

This table describes chapters related to bulk data loading with spreadsheets.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>Data Loading Using Spreadsheets</td>
<td>An overview of the available spreadsheet loaders and general setup requirements for spreadsheet usage.</td>
</tr>
<tr>
<td>25</td>
<td>Introduction to HCM Spreadsheet Data Loader</td>
<td>An overview of HCM Spreadsheet Data Loader, its processing, and its access requirements.</td>
</tr>
<tr>
<td>26</td>
<td>HCM Spreadsheet Data Loader Templates</td>
<td>How to create, edit, copy, export, and import spreadsheet templates.</td>
</tr>
<tr>
<td>27</td>
<td>Importing and Loading Data Using HCM Spreadsheet Data Loader</td>
<td>How to generate a spreadsheet from a template, populate the spreadsheet, import and load the data, and fix any errors.</td>
</tr>
</tbody>
</table>

## Postload Processing

This table lists chapters related to postload processing for bulk data loading.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>Postload Processing for HCM Data Loader</td>
<td>Processes that you must run after loading specific business objects using either HCM</td>
</tr>
</tbody>
</table>
## Chapter 1

### Introduction to Integrating with Oracle HCM Cloud

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>Purging Person Data from Test Environments</td>
<td>Processes to run in test environments to purge person and dependent data.</td>
</tr>
<tr>
<td>30</td>
<td>HCM Extracts and Diagnostics for HCM Data Loader and HCM Spreadsheet Data Loader</td>
<td>How to configure and run HCM extracts for HCM Data Loader. How to run the HCM Data Loader Error Analysis Report diagnostic test and the HCM Spreadsheet Data Loader diagnostic report.</td>
</tr>
</tbody>
</table>

## Best Practices

This table lists chapters that offer advice for your data loading efforts.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Best Practices for HCM Data Loader and HCM Spreadsheet Data Loader</td>
<td>Recommended approaches to data creation and maintenance.</td>
</tr>
</tbody>
</table>

## Loading Payroll

This table lists chapters associated with loading third-party data.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>Time, Absence, and Pension Data for Payroll</td>
<td>Loading third-party time entries, pension deductions, and absence information in XML format to Oracle Global Payroll.</td>
</tr>
</tbody>
</table>

## Extracting Data

This table describes chapters that discuss data extracts.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>Managing HCM Extracts</td>
<td>Using HCM Extracts to extract data from Oracle HCM Cloud.</td>
</tr>
<tr>
<td>34</td>
<td>Filtering and Delivering HCM Extracts</td>
<td>Filtering HCM Extracts data and selecting a delivery format.</td>
</tr>
</tbody>
</table>
Introduction to Integrating with Oracle HCM Cloud

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>35</td>
<td>Flow Patterns</td>
<td>Using flow patterns to complete extract reports and processes.</td>
</tr>
<tr>
<td>36</td>
<td>Benefits Data Extract</td>
<td>Setting up benefits plan carriers and extracting benefits enrollment information into an XML file for each carrier.</td>
</tr>
<tr>
<td>37</td>
<td>Talent Review and Succession Management Data Extract</td>
<td>The user entity, database items group, and extract definition type for the Talent Review and Succession Management data that you can extract using HCM Extracts.</td>
</tr>
</tbody>
</table>

Transferring Encrypted Data

This table describes chapters that relate to moving encrypted data.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>Transferring Encrypted Data</td>
<td>How to set up data encryption. How to upload and extract encrypted data.</td>
</tr>
</tbody>
</table>

Oracle Taleo Recruiting Cloud Service Integration

This table describes chapters that cover integration between Oracle Taleo Recruiting and Oracle HCM Cloud.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>Oracle Taleo Recruiting Cloud Service Integration</td>
<td>How integration between Oracle Taleo Recruiting Cloud Service and Oracle HCM Cloud is managed.</td>
</tr>
</tbody>
</table>

Web Services

This table lists chapters that discuss integrating web services with Oracle HCM Cloud.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>Web Services</td>
<td>Using web services to integrate web-based applications with Oracle HCM Cloud.</td>
</tr>
</tbody>
</table>
Inbound Integrations

This topic introduces ways of developing inbound integrations to Oracle HCM Cloud.

These include:

- HCM Data Loader
- HCM Spreadsheet Data Loader

HCM Data Loader

Use HCM Data Loader to load data in bulk from any source to Oracle HCM Cloud.

HCM Data Loader loads:

- Complex, hierarchical data
- Large volumes of data, such as complete system extracts
- Object history

You can use HCM Data Loader for both data migration and ongoing data maintenance. Major benefits of HCM Data Loader include support for:

- Most HCM business objects
- Incremental or partial data loading
- Four types of keys to identify records uniquely
- Configured descriptive flexfields and extensible flexfields, translation data, tree hierarchies, person images, and document-record attachments

HCM Data Loader provides:

- A comprehensive user interface for initiating data upload, monitoring progress, and reviewing errors
- A template file for each supported business-object hierarchy, which you can use as the basis for your own data files
- A user interface for reviewing the structure and attributes of every supported business-object hierarchy
- A web service that you can use to automate data upload

*Note:* The precursor of HCM Data Loader is HCM File-Based Loader. For information, see Overview: Transitioning from HCM File-Based Loader to HCM Data Loader (2206596.1) on My Oracle Support at https://support.oracle.com.

HCM Spreadsheet Data Loader

Use HCM Spreadsheet Data Loader to import data to the HCM Data Loader stage tables from spreadsheets. HCM Spreadsheet Data Loader supports most of the business objects that HCM Data Loader supports. It offers both ease of use and flexibility. For example, you can configure spreadsheet templates to suit business needs by omitting attributes, changing attribute order, changing attribute labels, and adding help text.

*Related Topics*

- Overview of HCM Data Loader
• Guidelines for Using HCM Spreadsheet Data Loader

Outbound Integrations

This topic introduces ways of developing outbound integrations from Oracle HCM Cloud. These include:

- HCM Extracts
- Oracle Business Intelligence Publisher (Oracle BI Publisher)

HCM Extracts

The main way to retrieve data in bulk from Oracle HCM Cloud is HCM Extracts, which is a tool for generating data files and reports.

HCM Extracts has a dedicated interface for specifying the records and attributes to be extracted. You:

- Identify records for extraction using complex selection criteria.
- Define data elements in an HCM extract using fast formula database items and rules.

You manage HCM Extracts either in the Data Exchange work area or using the Checklists interface in the Payroll work area. Alternatively, you can run extracts using the Flow Actions Service web service from outside Oracle HCM Cloud. This feature enables you to automate the outbound extract as part of an overall integration flow.

HCM Extracts Output Formats

Using the built-in integration with Oracle BI Publisher, you can generate extracts in various formats, including CSV, XML, Microsoft Excel, HTML, RTF, and PDF.

You can download the raw hierarchical XML from a completed extract using the Extract Actions Service web service. Alternatively, you can download the formatted BI report (for example, the CSV file) using the BI ScheduleService. In both cases, the output is streamed in a byte array. Therefore, any orchestration mechanism must handle this output, possibly by means of wrapper code.

You can also generate an HCM Extract without specifying an output format at the design stage. Subsequently, you can generate a BI report based on the extract using either Oracle BI Publisher or the BI ReportService. This technique uses a dedicated BI data model that interfaces with the generated HCM Extracts XML.

Oracle BI Publisher

Oracle BI Publisher supports both scheduled and unplanned reporting, based on either predefined Oracle Transactional Business Intelligence analysis structures or your own data models. You can generate reports in various formats. To use Oracle BI Publisher for outbound integrations, you generate reports in a format suitable for automatic downstream processing, such as XML or CSV.
Introduction to HCM Data Loader

Overview of HCM Data Loader

HCM Data Loader is a powerful tool for bulk-loading and maintaining data. The data can be from any source. You can use HCM Data Loader for data migration, ongoing maintenance of HCM data, and coexistence scenarios, where core HR data is uploaded regularly. This topic introduces the main features and advantages of HCM Data Loader.

Business-Object Support

You can load business objects for most Oracle Fusion Human Capital Management (HCM) products. These products include Oracle Fusion Absence Management, Compensation, Global Human Resources, Global Payroll, Performance Management, Profile Management, Talent Review and Succession Management, and Workforce Management. You can also update business objects using HCM Data Loader, regardless of how they were created.

Ease of Use

HCM Data Loader has a comprehensive user interface for loading data, monitoring progress, and reviewing any errors. It provides real-time information for all stages of its processing. HCM Data Loader’s user interface also includes detailed information about the component hierarchies and attributes of supported business objects.

You can load data from either delimited data (.dat) files or spreadsheets. You can load most supported business objects using either method. You can generate business-object templates to use as the basis of your own .dat files. For a selected business-object component, the template includes every attribute, including those for configured flexfields. When using spreadsheets, you define spreadsheet templates to suit business needs and generate spreadsheets from those templates.

Performance

HCM Data Loader supports multithreaded processing, which enables you to upload complete data extracts without severe performance impacts. References among objects that are processed on separate threads are managed automatically. You can set HCM Data Loader configuration parameters to optimize processing for your environment.

You can also perform partial or incremental loads to update existing objects, thereby minimizing the related processing.

Supported Key Types

HCM Data Loader supports all of the following key types for most business objects:

- Oracle Fusion GUID
- Oracle Fusion surrogate ID
- Source key
• User key

As user keys and source keys are supported, knowledge of Oracle Fusion internal IDs isn’t required.

Business-Object Features

HCM Data Loader supports most business-object features and requirements. For example, you can upload:

• Current and historical records for *date-effective objects*. You determine the amount of history to load.
• End-dated, terminated, or inactive records.
• Translated attributes in multiple languages. You specify the character set of the data file on the File Character Set configuration parameter for HCM Data Loader.
• Descriptive flexfields and extensible flexfields.
• Hierarchical tree data, such as organization and department trees.
• Attachments and pictures.
• Data from multiple sources. You can include source-system references in uploaded data.

Automation

You can initiate HCM Data Loader using a web-service call, which enables you to automate data upload.

HCM Data Loader and Implementation Scenarios

You can implement Oracle HCM Cloud in either full or coexistence mode. In a coexistence implementation, you use Oracle Fusion Talent Management or Oracle Fusion Compensation, but continue to use your existing HR applications. For both full and coexistence implementations, you implement Oracle HCM Cloud by performing the tasks that appear in your implementation project or for selected functional areas. This topic describes how HCM Data Loader supports these implementation types.

Full Implementations

If you’re performing a full implementation of Oracle HCM Cloud, then you can use HCM Data Loader to bulk-load any existing HCM data at appropriate stages in the implementation. Typically, you load each type of data once only for this type of implementation. Following successful upload, you manage your data in Oracle HCM Cloud.

Coexistence Implementations

In a coexistence implementation, you use Oracle Fusion Talent Management or Oracle Fusion Compensation, but maintain your existing HR applications. For this type of implementation, you:

• Move talent data permanently to Oracle HCM Cloud, which becomes the application of record for talent data.
• Upload other types of data, such as person records, periodically to Oracle HCM Cloud. The source system remains the application of record for this data.

To implement an HCM coexistence scenario, for any source system, you can use HCM Data Loader for data upload. Follow the general instructions provided for HCM Data Loader. Oracle doesn’t supply tools to extract data from your source systems. Work with your implementation partner or develop your own process for extracting the source data.

HCM Data Loader provides a Compensation Changes extract. You can extract assignment, bonus, and salary changes for a specified compensation run if you’re using Oracle Fusion Compensation in a coexistence scenario.

How Data Is Loaded Using HCM Data Loader

This topic outlines the process of loading data in bulk from .dat files using HCM Data Loader. The data can be from any source.

This figure provides a high-level summary of the process.

The steps of this process are:

1. You place a .zip file containing your .dat files on the Oracle WebCenter Content server.
2. You submit a request to HCM Data Loader to import and load the .zip file. For this step, you can use either the HCM Data Loader interface or the HcmCommonDataLoader web service.
3. HCM Data Loader decompresses the .zip file and imports individual data lines to its *stage tables*. In the stage tables, related data lines are grouped to form business objects. Any errors that occur during the import phase are reported on the HCM Data Loader interface.

4. HCM Data Loader calls the relevant logical object interface method (delivered in product services) to load valid objects to the application tables. Any errors that occur during the load phase are reported in the HCM Data Loader interface.

5. You review any errors from the import and load phases. You can perform this step either on the HCM Data Loader interface or using the HCM Data Loader Data Set Summary extract.

6. You correct errors from the import and load phases in your source data.

7. You load a new .zip file containing the corrected data to the WebCenter Content server.

You repeat this process from step 2 until all of the data is successfully loaded.

**Tip:** You can also correct load errors interactively on the HCM Data Loader interface and resubmit any corrected data from there.

**Related Topics**
- How Data Is Uploaded Using HCM Spreadsheet Data Loader

**Supported Key Types**

When planning to integrate data from one system to another, you must select key types to identify records uniquely. This topic introduces the key types that HCM Data Loader supports.

**What Keys Identify**

In HCM Data Loader, you use unique keys to identify:

- The record that you’re creating or updating.
- The parent of the record that you’re creating or updating.
  - The parent record may either be in the same data file or exist already in Oracle HCM Cloud.
- Any objects referenced by the record that you’re creating or updating.

**Supported Key Types**

HCM Data Loader supports the following key types, which you can use for all types of object references. When more than one key value exists for an object, key references are resolved in this order:

1. Oracle Fusion Globally Unique Identifier (GUID)
2. Source key
3. Oracle Fusion surrogate ID
4. User key

If you supply multiple key values, then no cross-validation occurs. For example, if you supply both a GUID and a source key, then the GUID is used and the source key is ignored. If the source key identifies a different record from the GUID, then no error is raised.
Tip: Whenever possible, use source keys for both creating and updating business objects.

Key-Type Features
The following table summarizes the main features of the available key types. The Create and Update columns indicate whether you can use the key type when creating and updating objects.

<table>
<thead>
<tr>
<th>Key Type</th>
<th>Create</th>
<th>Update</th>
<th>Held on Object</th>
<th>Type</th>
<th>Generated Automatically</th>
</tr>
</thead>
<tbody>
<tr>
<td>GUID</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Hexadecimal</td>
<td>Yes</td>
</tr>
<tr>
<td>Surrogate ID</td>
<td>No</td>
<td>Yes (see note)</td>
<td>Yes</td>
<td>Numeric</td>
<td>Yes</td>
</tr>
<tr>
<td>Source Key</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Alphanumeric</td>
<td>Conditionally (see note)</td>
</tr>
<tr>
<td>User Key</td>
<td>Yes</td>
<td>Yes (see note)</td>
<td>Yes</td>
<td>Alphanumeric</td>
<td>No</td>
</tr>
</tbody>
</table>

Note: You can use surrogate IDs when updating objects, but the IDs may not be readily available to Oracle HCM Cloud users. Default source keys are generated only if you don’t supply a source key when creating an object. You can’t use user keys alone when updating some objects because their values are subject to change.

Keys that aren’t held on the object exist in the Integration Key Map table.

Foreign-Object References
Oracle Fusion GUIDs and surrogate IDs are generated only when the associated record is successfully created in Oracle HCM Cloud. Source keys aren’t recognized in Oracle HCM Cloud until the associated record exists in Oracle HCM Cloud. Therefore, before you attempt to load data that references foreign objects, you must ensure that those foreign objects exist in Oracle HCM Cloud. For a new implementation, you’re recommended to load each business object separately and ensure that it loads successfully before importing the next. If you supply all business objects in the same .zip file, then HCM Data Loader loads them in order of dependency. In this case, a reference to a foreign object fails only if the referenced object failed to load.

Source Keys
A source key is the ID that identifies a business object in its source system. Source keys are supported only for integration-enabled business objects. You can use source keys when creating or updating business objects. You’re recommended to use source keys for all implementations but particularly for ongoing integrations, where data is maintained in the source system.

Source-Key Structure
A source key has the two components shown in this table.
Having a source-system owner component to the key enables you to upload data from multiple source systems for the same business objects. For example, you may have person data in both US and UK databases to combine in one Oracle HCM Cloud system. The source-system ID doesn’t have to be unique across both source systems. It must be unique only for the business-object component and source-system owner.

If you don’t have an obvious source-system ID for a record, then you can generate or derive one. For example, you could derive a source-system ID for a person address by concatenating the *person number* with the address type.

Source keys aren’t held against the created record. They exist in the Integration Key Map table.

### Source Keys for Date-Effective Business Objects

If you’re supplying date-effective history for an object, then you must supply the source system IDs for every date-effective record in the file. The values must be the same for each line of the date-effective history.

### Default Source Keys

If you provide no source key when creating an object, then the source key is created automatically. The default source-system owner is **FUSION** and the default source-system ID is the surrogate ID. You can update the source key later using the Source Key business object. In the Source Key business object, you supply the new source key and a reference to the object that you’re updating.

### Source-System Owner Values

The source-system owner value is validated against the HRC_SOURCE_SYSTEM_OWNER *lookup type*. You must add your source-system names to this lookup type before loading data using source-system references. Use the Manage Common Lookups task in the Setup and Maintenance work area to update HRC_SOURCE_SYSTEM_OWNER.

### User Keys

Most Oracle HCM Cloud business objects, regardless of whether they’re enabled for integration, have one or more attributes that make up a user key. The user key, which is also known as a natural key, is always visible on the user interface.

For example:

- The user key for an organization is the organization name.
- The user key for an area of responsibility is the responsibility name plus the *person number*. 
When to Use User Keys

User keys are part of the business object definition. They’re always required when you create a logical object, regardless of how you create it.

You can also use user keys when updating logical objects. However, the value of a user key can change, and some user-key attributes are translatable. For this reason, you can’t update some business objects if you supply only user keys to reference them. Therefore, whenever possible, you’re recommended to use source keys when updating objects.

User keys are recommended when:

- You’re referencing or maintaining an object that wasn’t created with a source key.
- The source-key value is unknown.

User Keys for Child Objects

When a business object is bound by another, the user key must include the user key for its parent. For example:

- Jobs are always part of a set. Therefore, job code alone doesn’t uniquely identify a job. Instead, the set code must be part of the user key for a job.
- Job grades apply to a specific job. Therefore, the user key for a job grade must include both the user key for the grade and the user key for the parent job. Therefore, the user key for the job grade is made up of the grade code, job code, and set code.

Oracle Fusion GUIDs and Surrogate IDs

Oracle Fusion Globally Unique Identifiers (GUIDs) and surrogate IDs are two of the four types of keys that HCM Data Loader supports. This topic describes when you can use GUIDs and surrogate IDs to identify business objects uniquely.

Oracle Fusion GUIDs

When you create an integration-enabled business object in Oracle HCM Cloud, a GUID is generated for the object automatically. As the GUID is generated when you create the business object, it’s suitable only for identifying business objects that already exist. You can’t supply a GUID when creating a business object. The GUID, a hexadecimal value, isn’t held on the created object but exists in the Integration Key Map table.

GUIDs are useful when you share changes with down-stream applications, such as third-party payroll providers. You extract the changes for the third-party payroll and provide the GUID as the unique key to the worker. When reporting any updates back to Oracle HCM Cloud, you can use GUIDs to identify the records to update.

Oracle Fusion Surrogate IDs

The surrogate ID, a numeric value, is the internal system identifier that’s generated when you create a business object. All objects, including those that aren’t integration-enabled, have surrogate IDs. The surrogate ID is held on the object. The ID is also used in the Integration Key Map table to provide a mapping for GUID and source-key values. As the surrogate ID isn’t generated until the object is created, you can’t use this value when creating objects. Access to surrogate IDs may also be limited. On-premises customers are more likely to have access to surrogate IDs than Oracle HCM Cloud customers.
Business-Object Structure

Each Oracle HCM Cloud business object is a hierarchy of business-object components. At the top of the hierarchy is the parent component, and beneath it may be child and grandchild components. Each business-object component has multiple attributes.

This figure shows a generic business-object structure.

For example, the Rating Model business object includes Rating Model, Rating Level, and Rating Category components. The Rating Model component is the parent of the other two components. Each component has attributes such as RatingName, RatingModelCode, RatingLevelCode, and so on.

The most complex business object supported by HCM Data Loader is the Worker object, where five levels exist in the object hierarchy. These range from the Worker component at the top to Assignment Work Measure, Assignment Manager, Assignment Grade Steps, and Assignment Extra Information at the bottom. By contrast, the Person Type object has only the Person Type component.

Terminology

This table defines the terms used by HCM Data Loader to refer to business objects.

<table>
<thead>
<tr>
<th>Term</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object or business object</td>
<td>Refers to the complete object, which is the parent component and all child components. For example, Grade and Worker are business objects.</td>
</tr>
</tbody>
</table>
### Loading Logical Objects

When you deliver multiple components for the same business object together, HCM Data Loader groups them and loads the complete logical object. It doesn’t process the components separately. If any component of the logical object fails validation, then the whole logical object is rejected. As HCM Data Loader loads only complete logical objects, you can be sure of exactly which data has been loaded. For example, when loading jobs, you know which jobs loaded successfully and which jobs failed to load.

### Integration-Enabled Objects

HCM Data Loader can load any integration-enabled business object. An integration-enabled business object supports these four key types:

- Oracle Fusion GUID
- Oracle Fusion surrogate ID
- Source key
- User key

An integration-enabled object has an entry in the Integration Key Map table, where its GUIDs and source keys exist.

HCM Data Loader can also load a few objects, such as Department Tree, that aren’t integration enabled. Typically, Oracle HCM Cloud doesn’t own such objects and they support neither source keys nor GUIDs.

**Tip:** You can review business-object details on the View Business Objects page. The Integration Keys Supported field on the Component Details tab for a selected component indicates whether the component is integration enabled.

### Related Topics

- Sources of Business-Object Information

### FAQs for Introduction to HCM Data Loader

**Why can't I access HCM Data Loader?**

You may not have the correct permissions. You must have the Human Capital Management Integration Specialist job role or privileges to access HCM Data Loader.
You define the name of the .zip file, which must be in alphanumeric characters (a-z, A-Z, and 0-9). The .zip file can be encrypted as a whole, but HCM Data Loader doesn’t support the encryption of individual data files.

**BlobFiles and ClobFiles Folders in the .zip File**

The only two folders that can appear in the .zip file are:

- BlobFiles, which is for binary large objects
- ClobFiles, which is for character large objects

You use these folders for files you want to load as attachments or into large objects. The file names of these attachments or images must be in alphanumeric characters (a-z, A-Z, and 0-9). The data type of the attribute that’s used to load your attachment or large-object data determines which folder to use. For example, the **File** attribute in the Document Record object is used for loading attachment files and has a data type of BLOB. Therefore, you place files to be loaded as attachments to document records in the BlobFiles folder.
This figure shows a .zip file structure when attachments are included.

![Figure showing .zip file structure with BlobFiles folder and files](image)

This figure shows the contents of the BlobFiles folder.

![Figure showing contents of BlobFiles folder](image)

### Data Files

Each HCM Data Loader data file is specific to a business object. You provide all components that you want to load for a business-object hierarchy in the same data file. For example, you provide Job, Job Valid Grade, Job Evaluation, Job Extra Information, and Job Legislative Extra Information components in Job.dat. This approach lets you see all data for a business object and supports the validation of parent references on all child components. It also simplifies the construction of the .zip file, as separate directories aren’t required and fewer data files have to be generated.

![Tip: You can generate data-file templates for all supported objects on the View Business Objects page. Each template defines the complete structure of the selected object.](image)

### Third Party Data Files

You can use third party .zip files to load data with HCM Data Loader, such as pay slips that were generated by a payroll application. The data file must conform to the requirements for the business object you want to load, and the .zip file must have been compressed with `java.util.zip`. If not, then you must extract the files and compress again using `java.util.zip` for the load to work properly.

### Related Topics

- How You Load Images, Attachments, and Large Strings
File Line Instructions and File Discriminators

Each line in a data file begins with an instruction, and most lines include a file discriminator. This topic introduces the available instructions and describes the file discriminators.

File Line Instructions

This table identifies the instructions that can appear in an HCM Data Loader data file.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Line Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>METADATA</td>
<td>Definition</td>
<td>Identifies the business-object component and the attributes of that component for which you’re including values in the data file</td>
</tr>
<tr>
<td>MERGE</td>
<td>Data</td>
<td>Provides data to be merged in Oracle HCM Cloud</td>
</tr>
<tr>
<td>DELETE</td>
<td>Data</td>
<td>Identifies a business-object component to be purged from Oracle HCM Cloud</td>
</tr>
<tr>
<td>SET</td>
<td>Control</td>
<td>Overrides aspects of the default processing for a file</td>
</tr>
<tr>
<td>COMMENT</td>
<td>Comment</td>
<td>Adds a comment to the data file</td>
</tr>
</tbody>
</table>

File Discriminators

METADATA, MERGE, and DELETE lines must include a file discriminator value, which identifies the business-object component to which the instruction applies. For example, the available file discriminators for the Job business object are shown in this table.

<table>
<thead>
<tr>
<th>Business-Object Component</th>
<th>File Discriminator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job</td>
<td>Job</td>
</tr>
<tr>
<td>Job Valid Grade</td>
<td>JobValidGrade</td>
</tr>
<tr>
<td>Job Evaluation</td>
<td>JobEvaluation</td>
</tr>
<tr>
<td>Job Extra Information</td>
<td>JobExtraInfo</td>
</tr>
<tr>
<td>Job Legislative Extra Information</td>
<td>JobLegislative</td>
</tr>
</tbody>
</table>
The file discriminator appears immediately after the instruction in a file line. HCM Data Loader templates for a business object include the file discriminator for each of the object’s components. This example shows instructions, the file discriminator, attribute names, and attribute values for the Job business object.

METADATA|Job|EffectiveStartDate|EffectiveEndDate|SetCode|JobCode|Name
MERGE|Job|1950/01/01|4712/12/31|COMMON|CFO|Chief Financial Officer
MERGE|Job|1950/01/01|2013/01/01|COMMON|PM|Manager Projects
MERGE|Job|2013/01/02|4712/12/31|COMMON|PM|Manager Projects
MERGE|Job|1950/01/01|4712/12/31|COMMON|VP_OF_HR|VP Human Resources
MERGE|Job|1950/01/01|4712/12/31|COMMON|PROG_MGR|Program Manager

In this example:
- The instruction, METADATA or MERGE, appears at the start of the line.
- The discriminator on all lines is Job.
- Attribute names appear in the METADATA line after the Job discriminator and separated by the vertical bar character (|).
- Attribute values appear in MERGE lines after the Job discriminator and separated by the vertical bar character (|).

The METADATA Instruction

The METADATA instruction in an HCM Data Loader data file identifies a business-object component and its attributes. The attributes are those for which you’re including values in the data file.

Instruction Format

The METADATA line has this format:

METADATA|<DISCRIMINATOR>|<ATTRIBUTE 1>|<ATTRIBUTE 2>|<ATTRIBUTE n>

For example:

METADATA|Job|SetCode|JobCode|JobFamilyName|JobName|EffectiveStartDate|EffectiveEndDate

Restrictions

These restrictions apply to the METADATA instruction:

- The METADATA line must appear before the MERGE or DELETE data line to which it relates.
- You can include multiple METADATA lines in a single data file. However, each line must be for a different business-object component. For example, in a Job.dat file, you can include METADATA lines for the Job and Job Valid Grade components. However, you can’t include two Job METADATA lines.

Validation

Each METADATA line must:

- Include a valid discriminator for the object that’s identified in the data file name. For example, in a Job.dat file, the METADATA line must include one of these discriminators: Job, JobGrade, JobEvaluation, JobExtrainfo, and JobLegislative.
- Be unique in the file. That is, you can’t provide multiple METADATA lines for the same discriminator in a single data file.
For example, having two METADATA lines for the JobGrade discriminator in the same data file is invalid.

- Include only valid attributes for the discriminator. Attribute names are case-sensitive, but they can appear in any order.
- Include the attributes for at least one of the supported key types. For example, if you’re using source keys to identify a record uniquely, then you must include SourceSystemOwner and SourceSystemId attributes in the METADATA line.

HCM Data Loader business-object templates include METADATA lines with all available attributes for each component of a business object.

### The MERGE Instruction

The MERGE instruction identifies a data line. You use the MERGE instruction whether creating or updating objects, and HCM Data Loader identifies the correct action.

You can supply either partial or complete objects using the MERGE instruction. The data supplied on the MERGE line is merged into existing Oracle HCM Cloud data. The order in which you specify MERGE lines isn’t necessarily the order in which HCM Data Loader processes them.

### Instruction Format

The MERGE line has this format:

```
MERGE|<DISCRIMINATOR>|<ATTRIBUTE 1>|<ATTRIBUTE 2>|<ATTRIBUTE n>
```

For example:

```
MERGE|Job|SE|Software Engineer|Common|2015/01/01|4712/12/31
```

### Restrictions

These restrictions apply to the MERGE line:

- You must not supply multiple MERGE lines for the same component in a single data file, unless the object is date-effective. For example, you can’t supply two MERGE lines, one to create a Person Ethnicity component and one to update it, in the same data file. As the component isn’t date-effective, these instructions would correct the data rather than update it. In such cases, you can supply only current data. Alternatively, you can create the record in one data file and update it in a different data file.

> **Note:** HCM Data Loader doesn’t process individual file lines, but groups related lines. This grouping works for date-effective records because the file lines are processed in order of effective start date.

- You can include MERGE and DELETE instructions in the same file, but not for the same logical record. For example, you can’t load multiple date-effective records for the same component, such as a single Job, using a mix of MERGE and DELETE instructions. However, you can provide both MERGE and DELETE instructions for components of a single logical object. For example, you can provide:
  - MERGE instructions for Worker and Person Name
Validation

Each MERGE line must:

- Be preceded by the METADATA line for the same discriminator.
- Contain the same attributes as the relevant METADATA line and in the same order.
- Not include values for attributes that aren’t in the METADATA line.
- Contain a unique identifier for itself, using any of the supported key types. In addition, each data line for a child component must include a unique reference to its parent. For example, GradeRateValue must include a unique reference to its associated GradeRate.

The DELETE Instruction

The DELETE instruction identifies a business-object component to purge from Oracle HCM Cloud. The component doesn’t have to have been loaded using HCM Data Loader.

Instruction Format

The DELETE instruction has this format:

```
DELETE|<DISCRIMINATOR>|<ATTRIBUTE 1>|<ATTRIBUTE 2>|ATTRIBUTE n>
```

For example:

```
METADATA|JobFamily|EffectiveStartDate|EffectiveEndDate|JobFamilyName
DELETE|JobFamily|2012/10/01|4712/12/31|Sales01
```

Restrictions

These restrictions apply to the DELETE instruction:

- You can delete a business object or a component of a business object. However, you can’t delete individual date-effective records.
- You can’t provide MERGE and DELETE instructions for the same record in a single data file. The order in which HCM Data Loader processes instructions is unpredictable.
- You can’t recover deleted records. Therefore, try to correct data rather than delete and recreate it.
- Not all business-object components can be deleted. For information about deletion support, see the business-object details available from the View Business Objects page in the Data Exchange work area.

The SET Instruction

The SET instruction is optional in a .dat file and has no discriminator. You use SET instructions to alter default processing for the file in which they appear. You can include any number of SET instructions in a .dat file, but they must appear before the first METADATA line. This topic describes when to use the SET instruction.
This table lists the SET instructions and their default values.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET PURGE_ FUTURE_ CHANGES Y</td>
<td>N</td>
</tr>
<tr>
<td>SET DISABLE_ POST_ PROCESS_ TASKS &lt;process&gt;</td>
<td>Not applicable</td>
</tr>
<tr>
<td>SET INVOKE_ POST_ PROCESS Y</td>
<td>N</td>
</tr>
<tr>
<td>SET FILE_ DELIMITER &lt;delimiter&gt;</td>
<td>Vertical bar (</td>
</tr>
<tr>
<td>SET FILE_ESCAPE &lt;escape character&gt;</td>
<td>Backslash ()</td>
</tr>
<tr>
<td>SET FILE_NEWLINE &lt;newline character&gt;</td>
<td>n</td>
</tr>
<tr>
<td>SET ENABLE_ AUDIT_DATA Y</td>
<td>N</td>
</tr>
<tr>
<td>SET PURGE_ AUDIT_DATA Y</td>
<td>N</td>
</tr>
<tr>
<td>SET CALCULATE_FTE Y</td>
<td>N</td>
</tr>
<tr>
<td>SET CREATE_DEFAULT_ WORKING_HOUR_PATTERN Y</td>
<td>N</td>
</tr>
</tbody>
</table>

**Tip:** The default values shown here are predefined for the enterprise. You can override these values for the enterprise using the Configure HCM Data Loader task in the Setup and Maintenance work area.

**SET Instruction Descriptions**

This section provides an alphabetic list of all SET instructions with their descriptions.

**SET CALCULATE_FTE**

To request automatic calculation of full-time equivalent (FTE) values for all worker assignments in a Worker.dat file, include the SET CALCULATE_FTE instruction. For example:

```
SET CALCULATE_FTE Y
```

**SET CREATE_DEFAULT_WORKING_HOUR_PATTERN**

To request automatic creation of a default working-hour pattern for all worker assignments in a Worker.dat file, include the SET CREATE_DEFAULT_WORKING_HOUR_PATTERN instruction. For example:

```
SET CREATE_DEFAULT_WORKING_HOUR_PATTERN Y
```
SET DISABLE_POST_PROCESS_TASKS

By default, these processes run automatically after you load workers:

- Refresh Manager Hierarchy
- Update Person Search Keywords

To prevent Refresh Manager Hierarchy from running automatically, you can include this SET instruction in the Worker.dat file:

```
SET DISABLE_POST_PROCESS_TASKS RefreshManagerHierarchy
```

To prevent Update Person Search Keywords from running automatically, you can include this SET instruction in the Worker.dat file:

```
SET DISABLE_POST_PROCESS_TASKS UpdatePersonSearchKeywords
```

By default, the FlattenPositionHierarchy process runs automatically after you load HCM position hierarchies. To prevent FlattenPositionHierarchy from running automatically, include this SET instruction in the PositionHierarchy.dat file:

```
SET DISABLE_POST_PROCESS_TASKS FlattenPositionHierarchy
```

Tip: To prevent postload processes from running automatically, you can use the SET INVOKE_POST_PROCESS instruction instead.

SET ENABLE_AUDIT_DATA

Audit of objects loaded using HCM Data Loader is enabled or disabled for the enterprise using the Enable Audit Data configuration parameter. You can override the enterprise setting for individual .dat files by including this SET instruction. For example, if audit is enabled for the enterprise, then you can disable it for an individual .dat file.

SET FILE_DELIMITER

Use this instruction to change the default file delimiter, which is the vertical bar (|). For example, to change it to a comma, include this instruction:

```
SET FILE_DELIMITER ,
```

The new value can be up to 10 characters.

SET FILE_ESCAPE

Use this instruction to change the default file escape character, which is the backslash (\). For example, to change it to the equal sign (=), include this instruction:

```
SET FILE_ESCAPE =
```

The new value can be up to 10 characters.

SET FILE_NEWLINE

Use this instruction to change the default file newline character, which is the character \n. For example, to change it to the string newline, include this instruction:

```
SET FILE_NEW_LINE newline
```

The new value can be up to 10 characters.
**SET INVOKE_POST_PROCESS**

By default, these processes run automatically after you load workers:

- Refresh Manager Hierarchy
- Update Person Search Keywords

By default, the **FlattenPositionHierarchy** process runs automatically after you load HCM position hierarchies.

To prevent such processes from running automatically, you can include this SET instruction in the relevant .dat file:

```
SET INVOKE_POST_PROCESS N
```

**SET PURGE_AUDIT_DATA**

Use this instruction to purge the audit data for the objects in a .dat file. This feature is intended primarily to remove the audit records for personally identifiable information in worker records.

You can’t include both of the following instructions in a single .dat file:

- `SET ENABLE_AUDIT_DATA Y`
- `SET PURGE_AUDIT_DATA Y`

If auditing of loaded objects is enabled for the enterprise, then you must disable it in any .dat file where you’re purging audit data. That is, you include the following instructions:

- `SET ENABLE_AUDIT_DATA N`
- `SET PURGE_AUDIT_DATA Y`

**SET PURGE_FUTURE_CHANGES**

When you create business objects containing date-effective components, you include this instruction at the start of the file:

```
SET PURGE_FUTURE_CHANGES Y
```

This instruction sets the date-effective maintenance mode to Replace. In this mode, which is the default mode, the contents of the .dat file replace any existing data.

When you update business objects containing date-effective components, you include this instruction at the start of the file:

```
SET PURGE_FUTURE_CHANGES N
```

This instruction sets the date-effective maintenance mode to Retain. In this mode, any existing future-dated changes are retained.

**Related Topics**

- How You Manage Reserved Characters
- Overview of Loading Date-Effective Objects
- Enable Audit for Oracle HCM Cloud Business Objects
- HCM Data Loader Configuration Parameters
Generate and Use Template Files

HCM Data Loader provides a template file for each supported business-object hierarchy. The template contains:

- A COMMENT line, which identifies the business object, its version, and the file-creation date.
- A METADATA line for each component of the business-object hierarchy that you can load for the business object. The METADATA line includes every attribute of the component, including environment-specific configured flexfield attributes.

You can generate template files and use them as the basis of your own data files. The topic describes how to generate and use template files.

Generate a Single Template File

Follow these steps:

1. Open the Data Exchange work area and select the View Business Objects task on the Tasks panel tab.
   By default, the View Business Objects page lists all business objects that HCM Data Loader supports. You can sort the list by product area, business object, or load order, for example. You can also retrieve a specific object by entering its name in the Business Object query-by-example field and pressing Enter.

2. If a template already exists for the object, then you can click the icon in the File column to download it. The Last Refreshed column shows when the object definition was last updated and the template was last generated.
   Alternatively, you can update the object definition before you download the template, as follows:
   a. Select the object row in the business objects table.
   b. Click Refresh Object.
   c. In the Schedule Request dialog box, click Submit. Click OK to close the confirmation message.
   When you click Submit, you start the process that updates the object definition and generates the latest template file.

   ⚠️ Tip: To monitor progress, click View Process Results to open the View Process Results page.

3. On the View Business Objects page, click the Refresh icon.
   Check the value in the Last Refreshed column to confirm that the template has been updated.

4. Click the icon in the File column to download the template.

   ⚠️ Tip: You can download the complete object definition for offline use. To download the object definition, click the icon in the Download column. The spreadsheet that you download is documentation for the object, not a template file.

Generate Template Files for All Business Objects

Follow these steps to generate template files for all business objects in a product area or all business objects:

1. Click the Refresh All Objects button on the View Business Objects page.
2. In the Schedule Request dialog box, you can either select a product area or leave Product Area set to All.
3. Click **Submit**.

4. When the generation process completes, you can download templates individually. Click the icon in the **File** column for an object on the View Business Objects page. Alternatively, follow these steps to download a single .zip file of the generated templates:
   - Click View Process Results.
   - On the View Process Results page, click the icon in the **File** column for your process.

### Use Template Files

Template files identify the names of all available attributes of a business object, including those of environment-specific attributes, such as flexfield segments. Don’t simply copy the METADATA lines from the template file and use them in your own data files. Instead, include in your data files:

- METADATA lines for the components that you want to load
- Attribute names for the values you want to supply

You can improve the speed and efficiency of the import and load processes by including only the attributes for which you’re supplying data. Remove METADATA lines and attribute names that you don’t plan to use.

HCM Data Loader validates every attribute name on every METADATA line. It also identifies any potential dependencies on other business objects in the same .zip file by reviewing the attributes supplied in the METADATA lines. For example, the Job.dat file may contain a METADATA line for the Job Valid Grade component with the **Grade ID** attribute. In this case, HCM Data Loader assumes that the Job.dat file has a potential dependency on grades.

### FAQs for Data File Instructions and Delivery

**How should I deliver data when creating objects?**

Deliver one object per .zip file. For example, create one .zip file for jobs, one for workers, and so on. Correct any errors before loading the next object to avoid data-reference errors.

**How should I deliver data for incremental updates?**

Supply only changed and required attribute values. Deliver all business-object files in the same .zip file. HCM Data Loader processes them in the correct order and loads referenced data before the data that references it.

**Can I include comments in a .dat file?**

Yes. Use the **COMMENT** instruction to add a comment to a .dat file.

For example:

```
COMMENT Data for Business Object: Job Version: V2 Created on: 25-03-2017
```

You can include any number of **COMMENT** instructions and they can appear anywhere in the file. They have no effect on processing.
Preparation to Use HCM Data Loader

Sources of Business-Object Information

To load a business object or component successfully using HCM Data Loader, you must understand its structure, supported key systems, data formats, and required attributes. You must also understand any business rules that apply to the object. This topic identifies where you can find this information.

Business-Object Details

When you click the name of a business object on the View Business Objects page, you access information about the object. This information includes:

- The component hierarchy of the object.
- Component details for the currently selected component. These include:
  - Its parent component, file discriminator, .dat file, and translation object
  - Whether it supports integration keys
  - Its supported actions, for example, whether you can delete the component
  - Whether it’s date effective and, if so, whether you can change the component’s first start date and last end date
- Attribute details for each attribute of the currently selected component. These include:
  - Its name, type, and length
  - Whether it’s required always or for new objects only
  - Whether it’s a key value or can be used as a key value and, if so, the key type
  - For attributes that are keys of type surrogate ID, parent surrogate ID, or foreign-object reference, whether integration keys are also supported
  - The names of any user-key attributes that you can use in place of surrogate IDs
  - Names of lookup types, for attributes that are lookup-validated

If you can load flexfields for the component, then the Flexfield Attributes tab appears. For descriptive flexfields and extensible flexfields, you can see the attributes for a selected flexfield and context. For extensible flexfields, category also appears. When a flexfield context supports multirow entries, one or more flexfield attributes are identified as user keys to reference a single record uniquely. These attributes are identified in the User Key column on the Flexfield Attributes tab. This column doesn’t appear if the flexfield doesn’t support multirow entries.

You can refresh business objects using the Refresh Object icon and the Refresh All Objects button. When you refresh an object:

- Attribute and flexfield information for the selected business objects is updated.
- Up-to-date templates of METADATA lines are generated for the selected business objects.
You’re recommended to refresh all objects after an update. The refresh ensures that any changes from the update appear on the View Business Objects page and in object templates. If no details exist for a selected object when you view it, then you’re prompted to refresh the object.

You can download the complete definition of an object for offline use by clicking the icon in the Download column for the object. The default file name of the downloaded spreadsheet is GenericBusObjDetails.xlsx, but you can change the name when you save the file.

Tip: Whenever you open a downloaded spreadsheet, you’re invited to sign in again. If you sign in, then the spreadsheet is updated automatically with latest information as you select each tab. These updates overwrite any changes that you may have made to the spreadsheet. If you prefer to keep your changes, then don’t sign in.

Business Rules

Business rules for many objects are included in the Integrating with Oracle HCM Cloud guide. For objects that aren’t described in the Integrating with Oracle HCM Cloud guide, see HCM Data Loader: Business Object Documentation (2020600.1) on My Oracle Support at https://support.oracle.com. This document also provides attribute descriptions and example .dat files for some business objects.

Example .dat files for many business objects are also available on Cloud Customer Connect. Select Navigator > Others > Cloud Customer Connect. After signing in, select the HCM Cloud Integrations forum on the Human Capital Management page.

Other documents on My Oracle Support provide information about business objects for specific localizations. Many are linked from the document All White Papers for Oracle Fusion HCM (1504483.1). For example:

- Oracle Fusion HCM: HCM Address Validation (2140848.1)
- Oracle Fusion HCM: Person Name Validation (2146270.1)
- Validating National Identifiers in Oracle Fusion Human Capital Management (2159758.1)

Related Topics

- HCM Data Loader: Business Object Documentation (2020600.1)

HCM Data Loader Configuration Parameters

The HCM Data Loader configuration parameters control the HCM Data Loader and HCM Spreadsheet Data Loader import and load processes. The default settings of the parameters suit most requirements. However, you can override these settings to suit the configuration of your environment and the amount of data being loaded. Users can also override most enterprise settings for individual load processes.

In the Setup and Maintenance work area, use the following to review or override the predefined settings for the enterprise:

- Functional Area: HCM Data Loader
- Task: Configure HCM Data Loader

On the Configure HCM Data Loader page, select the HCM Data Loader tab. You can filter the parameters by these category values:

- Availability
- Diagnostics
Parameters that are specific to a business object appear in a separate section. This topic describes each of the configuration parameters within its category. Object-specific parameters are described following the category sections.

**Note:** HCM Spreadsheet Data Loader has its own versions of some configuration parameters, which you manage on the HCM Spreadsheet Data Loader tab. For these parameters, you can set different default values for HCM Data Loader and HCM Spreadsheet Data Loader. Users can override the HCM Spreadsheet Data Loader default values in spreadsheet templates.

### Availability Parameters

**Enable Audit Data**

Specifies whether audit data is captured for business objects loaded using HCM Data Loader. If you set this parameter to Yes, then audit must also be enabled for Oracle Fusion Applications.

You can set this parameter separately for HCM Data Loader and HCM Spreadsheet Data Loader.

Default Value: No

**Enable Load Group Size Calculation**

Specifies whether the load-group size is to be calculated automatically. Set to Yes to enable the optimal load-group size to be calculated based on the number of objects to process and the available concurrent threads. When Enable Load Group Size Calculation is Yes, the Load Group Size parameter doesn’t appear on the Schedule Request page. Also, the default value of the Load Group Size parameter is ignored.

Default Value: Yes

**Enable UI Message Translation**

Specifies whether messages on the Import and Load Data page are to appear in the language of the signed-in user. Set to Yes to enable translated messages. If this parameter is set to No, then messages appear in the language of the user who submitted the import and load request.

Default Value: No

**Generate Business Object Mapping Files**

Specifies one of three options for generating Business Object Mapping Files. The mapping file describes the HCM Data Loader shape and you can import this into Oracle Integration Cloud to define transformation and mappings rules.

The default value of No means you haven’t enabled the option to generate mapping files.

The Custom only value lets you generate mapping files for business object dat files you upload to HCM Data Loader. Only the attributes included in the uploaded file will be included in the mapping file. Under the Parameters section, select the Generate Objects Mapping File feature. Refresh the Business Objects task bar until the Download Mapping File link activates and then you can retrieve the map.

A parameter value of Yes lets you request a mapping file from the Import and Load Data interface, just like Custom only. In addition, on the View Business Objects page, you can Refresh an individual object and it will automatically generate a mapping file for you. Refresh the page until Last Refreshed indicates today’s date. Download the mapping file using the link under Integration Mapping File.
Default Value: No

**HCM Data Loader Scope**

Specifies whether HCM Data Loader is used for all data loading.

From Release 10, the default value of the **HCM Data Loader Scope** parameter is **Full** for new customers only. You can’t change this setting.

For existing customers upgrading from Release 9, the default value of this parameter is **Limited**. If you set it to **Full**, then you can’t change the setting back to **Limited**.

Default Value: Full

**Purge Person Enabled Key**

Key value supplied by Oracle Support to enable the **Purge Person Data in Test Environments** process.

You can’t enable and run this process in production environments.

Default Value: None

**Diagnostics Parameters**

**Complex Error Stack Trace Occurrences by Thread**

Maximum number of complex error message occurrences on a processing thread for which stack trace is recorded.

Default Value: 5

**Data Error Stack Trace Occurrences by Thread**

Maximum number of data error message occurrences on a processing thread for which stack trace is recorded.

Default Value: 2

**File Definition Parameters**

**Data Set Name**

For data loads initiated by web service, specifies whether the Oracle WebCenter Content document title or the .zip file name is used as the data set name.

Default Value: File name

**File Character Set**

Character set used for business object and attachment files.

Default Value: UTF-8

**File Delimiter**

A value of up to 10 characters used to separate attribute names on METADATA lines and attribute values on data lines.

Default Value: Vertical bar (|)

**File Encryption**

Default file encryption.
Default Value: None

**File Escape Indicator**

A value of up to 10 characters used to indicate that the next character is to be ignored on data lines.

Default Value: Backslash (\)

**File New Line Indicator**

A value of up to 10 characters used to indicate a new line in an attribute value. This value must be preceded by the file escape indicator.

Default Value: n

**Flexfields Enabled**

Controls whether flexfield data can be loaded.

Default Value: Yes

**Purge Audit Data**

Specifies whether audit data is to be purged automatically for objects successfully updated or deleted.

Default Value: No

> **Note:** You can’t override the default value for the enterprise. You can include a SET instruction in individual .dat files to purge audit data for the objects in the files.

### Performance Parameters

**Days to Retain Data Sets Before Archiving**

The number of days since a data set was last updated before it can be automatically backed up in archive stage tables. Qualifying data sets are backed up only if the number of data lines in the stage tables exceeds the **Maximum Data Lines Before Archiving** value.

Default Value: 5

**Days to Retain Data Sets Before Deleting**

The number of days since a data set was last updated before it’s automatically and permanently deleted from the stage tables.

Default Value: 30

**Environment Configuration**

Value made up of 1 or 2 letters followed by a number in the range 0 through 99 that’s supplied by Oracle Support. HCM Data Loader uses this value to calculate the maximum number of concurrent threads that you can use when loading data. The resulting maximum value is used to validate the value that you specify:

- For the **Maximum Concurrent Threads for Load** parameter on the Configure HCM Data Loader page
- When importing and loading data on the Import and Load data page
- When importing and loading data using a web service
In these cases, if the **Maximum Concurrent Threads for Load** value is higher than the calculated maximum value, then an error is raised.

> Note: You raise a service request (SR) to obtain the value of this parameter for your environment.

If you leave the **Environment Configuration** parameter blank, then the default value of the calculated maximum is 1000.

Default Value: None

**Load Group Size**

Number of business objects processed as a single unit of work by a single thread.

Default Value: 100

**Maximum Concurrent Threads for Import**

Maximum number of threads to run concurrently when importing data to the stage tables.

You can set this parameter separately for HCM Data Loader and HCM Spreadsheet Data Loader.

Default Value: 8

**Maximum Concurrent Threads for Load**

Maximum number of threads to run concurrently when loading data from the stage tables to the application tables.

This parameter is used in conjunction with the **Load Group Size** parameter. Only the threads required to load your objects based on the **Load Group Size** parameter are created.

Default Value: 8

**Maximum Data Lines Before Archiving**

The number of data lines that can be held in the stage tables before data sets are automatically backed up.

Default Value: 10000000

**Transfer Group Size**

Number of file lines processed as a single unit of work by a concurrent thread.

Default Value: 100000

**Scheduling Default Parameters**

**Date-Effective Update Mode**

Specifies whether future-dated records should be purged when date-effective objects are loaded.

You can set this parameter separately for HCM Data Loader and HCM Spreadsheet Data Loader.

Default Value: Replace

**Delete Source File**

Specifies whether to delete source files from the Oracle WebCenter Content server once they have been processed.

Default Value: Yes

**Disable Post Process Tasks**
This parameter is read-only. You can disable individual postload processes by including a SET instruction in the relevant Worker.dat or Position.dat file.

Default Value: None

**File Action**

Default file-processing action.

**Import and load** means that valid objects are loaded automatically once import completes. **Import only** means that your data is imported to the stage tables but load isn't initiated.

Default Value: Import and load

**Initiate Business Object Post Processing**

Specifies whether processes registered to run automatically after an object is loaded should run.

You can set this parameter separately for HCM Data Loader and HCM Spreadsheet Data Loader.

Default Value: Yes

**Maximum Percentage of Import Errors**

Percentage of file lines in error that can occur in a business object before the import process stops for the object.

Default Value: 100

**Maximum Percentage of Load Errors**

Percentage of business-object instances in error that can occur for a business object before the load process stops.

You can set this parameter separately for HCM Data Loader and HCM Spreadsheet Data Loader.

Default Value: 100

**Business Object Parameters: Worker**

You can set these parameters separately for HCM Data Loader and HCM Spreadsheet Data Loader.

**Calculate Worker Full-Time Equivalent**

Specifies whether the full-time equivalent (FTE) value should be calculated automatically for all worker assignments in a Worker.dat file.

Default Value: No

**Create Worker Default Work Hour Pattern**

Specifies whether a default working-hour pattern should be created automatically for all worker assignments in a Worker.dat file.

Default Value: No

**Related Topics**

- The SET Instruction
HCM Data Loader Scope

The HCM Data Loader Scope parameter determines which business objects you can load using HCM Data Loader and which other data loaders you can use. The parameter has two possible values, as shown in this table.

<table>
<thead>
<tr>
<th>HCM Data Loader Scope Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limited</td>
<td>You can use HCM Data Loader to load only those business objects that aren’t supported by its predecessor, HCM File-Based Loader.</td>
</tr>
<tr>
<td>Full</td>
<td>You can use HCM Data Loader to load all supported business objects. You can’t use HCM File Based Loader, which is disabled.</td>
</tr>
</tbody>
</table>

For customers who upgraded from an earlier release to Release 10, the default value of this parameter is Limited. While you continue to use HCM File-Based Loader, leave HCM Data Loader Scope set to Limited. However, you should plan to use HCM Data Loader in place of HCM File-Based Loader. To ensure complete business-object support when using HCM Data Loader, you set HCM Data Loader Scope to Full.

For new customers, the default value of this parameter has been Full since Release 10.

> Note: You can’t change HCM Data Loader Scope from Full to Limited.

Restricted Objects

If HCM Data Loader Scope is set to Limited, then you can’t load the objects shown in the following table using HCM Data Loader. You can load them using HCM Data Loader only if HCM Data Loader Scope is set to Full.

<table>
<thead>
<tr>
<th>Module</th>
<th>Business Object</th>
<th>Business Object Translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absence Management</td>
<td>Person Absence Entry</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Absence Management</td>
<td>Person Entitlement Detail</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Compensation</td>
<td>Salary</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Compensation</td>
<td>Salary Basis</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Global HR</td>
<td>Actions</td>
<td>Actions Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Action Reasons</td>
<td>Action Reasons Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Content Item</td>
<td>Content Item Translation, Content Item Rating Description Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Content Items Relationship</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Module</td>
<td>Business Object</td>
<td>Business Object Translation</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>Global HR</td>
<td>Department Tree</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Global HR</td>
<td>Department Tree Node</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Global HR</td>
<td>Education Establishment</td>
<td>Education Establishment Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Grade</td>
<td>Grade Translation, Grade Step Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Grade Ladder</td>
<td>Grade Ladder Translation, Step Rate Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Grade Rate</td>
<td>Grade Rate Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Job</td>
<td>Job Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Job Family</td>
<td>Job Family Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Location</td>
<td>Location Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Organization</td>
<td>Organization Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Person Contact</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Global HR</td>
<td>Person Contact Relationship</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Global HR</td>
<td>Position</td>
<td>Position Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Rating Model</td>
<td>Rating Model Translation, Rating Category Translation, Rating Level Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Talent Profile</td>
<td>Talent Profile Translation</td>
</tr>
<tr>
<td>Global HR</td>
<td>Worker</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Global Payroll</td>
<td>Element Entry</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

If you try to load data for a business object that isn’t supported in **Limited** mode, then your whole data set fails to process.
Using HCM Spreadsheet Data Loader

The version of HCM Spreadsheet Data Loader that you can use depends on the value of the HCM Data Loader Scope configuration parameter. If HCM Data Loader Scope is set to:

- **Full**, then you can use HCM Spreadsheet Data Loader to load most objects that HCM Data Loader supports. All HCM Spreadsheet Data Loader functions are available. You use these tasks in the Data Exchange work area:
  - Manage Spreadsheet Templates
  - Run Spreadsheet Data Loader

- **Limited**, then you can use the version of HCM Spreadsheet Data Loader that was available in Release 11 and earlier. This version provides a static list of predefined spreadsheets. You use these tasks in the Data Exchange work area:
  - Initiate Spreadsheet Load
  - Load Spreadsheet Data

How You Define Referenced Objects

Business objects that you’re loading may reference a few business objects that you can’t load using HCM Data Loader because they’re not integration enabled. You define or review these objects in the target Oracle HCM Cloud environment before you load data that references them. You may have performed this step during implementation of Oracle HCM Cloud. This topic identifies how you define these referenced objects.

This table identifies the main objects of this type. It lists the tasks that you use to manage them and the functional area to which each task belongs in the Setup and Maintenance work area.

<table>
<thead>
<tr>
<th>Business Object</th>
<th>Functional Area</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Reference Data Set</td>
<td>Enterprise Profile</td>
<td>Manage Reference Data Sets</td>
</tr>
<tr>
<td>Assignment Status Type</td>
<td>Workforce Information</td>
<td>Manage Assignment Status</td>
</tr>
<tr>
<td>Business Unit</td>
<td>Organization Structures</td>
<td>Manage Business Unit</td>
</tr>
<tr>
<td>Content Type</td>
<td>Workforce Profiles</td>
<td>Manage Profile Content Types</td>
</tr>
<tr>
<td>Currency Code</td>
<td>Financial Reporting Structures</td>
<td>Manage Currencies</td>
</tr>
<tr>
<td>Element Type</td>
<td>Elements and Formulas</td>
<td>Manage Elements</td>
</tr>
<tr>
<td>Legal Entity</td>
<td>Legal Structures</td>
<td>Manage Legal Entity</td>
</tr>
<tr>
<td>Official Language Code</td>
<td>Application Extensions</td>
<td>Manage Languages</td>
</tr>
<tr>
<td>Profile Type</td>
<td>Workforce Profiles</td>
<td>Manage Profile Types</td>
</tr>
</tbody>
</table>
### Foreign-Object References

When you refer to these objects from objects that you’re loading, you use their user keys. (Alternatively, you can use their surrogate IDs, if available.) HCM Data Loader provides business-object documentation for all supported objects. This documentation identifies the user key that you can use for foreign-object references. For example, the Position object includes a reference to the Business Unit object, which isn’t integration enabled. The `position` documentation identifies the business-unit name as its user key. Therefore, when loading a Position object, you can refer to the associated *business unit* using the business-unit name. To see the supplied documentation for a business object, click the object name on the View Business Objects page.

### How You Review Lists of Values

The permitted values of many object attributes are defined in lists of values. Some lists of values are predefined and can’t be updated in any way. Others contain some values, but you can also add your own. In some cases, you can edit or remove predefined values. In Oracle HCM Cloud, lists of values are defined as *lookup types*. You’re recommended to review the predefined lookup types and make any updates before you attempt to load data that uses them. You may have completed this process during implementation of Oracle HCM Cloud.

### Reviewing Lookup Types

To manage lookup types, perform relevant tasks in the Setup and Maintenance work area. For example, the Workforce Information functional area contains tasks for managing many lookup types, including person lookups, termination lookups, document lookups, and checklist lookups. To manage person lookup types, perform the **Manage Person Lookups** task. On the Manage Person Lookups page, select a lookup type to edit. The list includes:

- Address types
- Contact relationships
- Ethnicity
- Highest education level
- Honors
- Marital status
- Military rank
- Phone types
- Religion

Ensure that the defined lookup values support the data that you’re planning to load.

**Related Topics**

- Overview of Lookups
Define the Source-System Owner

If you plan to use source keys, then you must add your source-system owner values to the HRC_SOURCE_SYSTEM_OWNER lookup type before you load data. You must have the Application Implementation Consultant or Human Capital Management Application Administrator job role or privileges to perform this task.

Use the Manage Common Lookups task in the Setup and Maintenance work area. Follow these steps:

1. On the Manage Common Lookups page, search for the lookup type HRC_SOURCE_SYSTEM_OWNER.
2. In the search results, select the lookup type to display its lookup codes.
3. In the Lookup Codes section of the page, click the New icon to add a lookup-code row.
4. Complete the fields in the row. Ensure that the new code is enabled and the dates are valid for the data that you're loading.
5. Repeat steps 3 and 4 for additional source-system owner values.
6. Click Save and Close.

Related Topics
- Source Keys

Guidelines for Preparing to Load Workers

Before you load Worker objects, you must decide:

- Whether you want user accounts to be created automatically for those workers.
- Whether user credentials are to be sent automatically to new users.
- Which roles are to be provisioned automatically to new users. User accounts created without at least one role are automatically suspended.

This topic describes each of these decisions.

Creating User Accounts Automatically

The enterprise option User Account Creation controls whether user accounts are created automatically for new workers, regardless of how those workers are created. If user accounts are created automatically in your enterprise, then you can prevent accounts being created for individual workers. To prevent account creation, include the GeneratedUserAccountFlag attribute of the User Information component and set it to N.

Note: If User Account Creation is set to None, then you can’t override it for individual workers.

You can supply a user name in the User Information component of the Worker object that you upload. Otherwise, user names are in the default format for the default user category, as specified on the Security Console. The default user-name format is the primary work email. Any changes that you make to the default format for a user category apply to all new users in the category, regardless of how they're created.

Tip: New users are added to the default user category. You can move an existing user to a different user category by setting the UserCategory attribute of the User object.
Sending User Credentials to New Users

If you create user accounts automatically for uploaded workers, then you can notify the users automatically of their user names and passwords. Set the **SendCredentialsEmailFlag** attribute of the User Information component to **Y** for any worker who should receive this mail. **SendCredentialsEmailFlag** is set to **N** by default.

If you set **SendCredentialsEmailFlag** to **Y** for any worker, then you must ensure that a valid notification template is enabled for the default user category for this event. Two predefined templates exist:

- The **New Account Template** is for notifying the user.
- The **New Account Manager Template** is for notifying the user’s manager.

You can also create notification templates. Notification templates are managed on the Security Console. Any changes made to notification templates apply to the user category.

Provisioning Roles to New Users

When a user account is created automatically for a worker, roles are provisioned automatically to the user as specified by current role-provisioning rules. Confirm that appropriate role mappings exist for users created by bulk upload.

❄ **Note:** All user accounts must have at least one role to remain active. If appropriate role mappings don’t exist and you’re not loading roles for manual assignment, then the new user account is immediately suspended. To avoid this automatic suspension of the account, define role mappings for workers before you load those workers.

*Related Topics*
- Role Mappings

How You Test the HCM Data Loader Process Flow and Connections

In a new environment, you may want to test HCM Data Loader end-to-end processing without creating unwanted data. Once you're sure that HCM Data Loader processing and connections are working as expected, you can load your own data with confidence. This topic describes how to test HCM Data Loader end-to-end processing.

Testing End-to-End Processing

HCM Data Loader provides:

- A process, **Test HCM Data Loader Process Flow and Connections**
- An object, Example Object

When you submit the **Test HCM Data Loader Process Flow and Connections** process, set the **Action** parameter to **Test HCM Data Loader**. The process:

1. Generates a file of MERGE instructions for the Example Object
2. Adds the ExampleObject.dat file to the ExampleObject||<date and time>||.zip file
3. Uploads the .zip file to the hcm/dataloader/import directory on the Oracle WebCenter Content server
4. Initiates HCM Data Loader to import and load the file
5. Provides a log file

You can review progress and results on the Import and Load Data page.

The Example Object

When you run the **Test HCM Data Loader Process Flow and Connections** process, the Example Object data is saved to its own tables. No data is saved to the application tables. You can review the structure and attributes of the Example Object on the View Business Objects page in the Data Exchange work area. The example object:

- Has child and grandchild components and a translation object
- Supports all key types
- Supports Create and Delete actions
- Has attributes of most data types
- Has lookup-validated attributes

You can generate a template file for the Example Object.

**Note:** Although an Example Object Translation object exists, no translation data is generated when you run the **Test HCM Data Loader Process Flow and Connections** process.

Deleting the Test Data

You're recommended to run the **Test HCM Data Loader Process Flow and Connections** process periodically to delete the test data and associated **data sets**. When you submit the process, set the **Action** parameter to **Delete test data**. In this case, the process generates a file of DELETE instructions, adds it to the DeleteExampleObject||<date and time>||.zip file, and uploads it.

How You Audit Objects Loaded in Bulk

You can audit changes made to objects when you bulk load them using either HCM Data Loader or HCM Spreadsheet Data Loader. When audit is enabled, the creation, update, and deletion of objects are audited. This topic describes how to enable and manage audit for bulk-loaded objects.

Enabling Audit for Bulk-Loaded Objects

Auditing is optional. To audit bulk loading of objects, you must:

1. Perform the **Manage Audit Policies** task in the Setup and Maintenance work area to select objects and their attributes for audit.
2. Enable auditing in general on the Manage Audit Policies page.
3. Enable auditing of bulk-loaded data, either for the enterprise or in individual .dat files and spreadsheet templates.
Enabling Audit for HCM Data Loader

The **Enable Audit Data** parameter on the HCM Data Loader parameters tab of the Configure HCM Data Loader page controls audit for the enterprise. By default, **Enable Audit Data** is set to **No**, but you can override that setting. Alternatively, you can enable audit selectively by including the following SET instruction in relevant .dat files:

```
SET ENABLE_AUDIT_DATA Y
```

Enabling Audit for HCM Spreadsheet Data Loader

The **Enable Audit Data** parameter on the HCM Spreadsheet Data Loader parameters tab of the Configure HCM Data Loader page controls audit for the enterprise. By default, the **Enable Audit Data** parameter is set to **No**, but you can override that setting. Alternatively, you can enable audit selectively by setting the **Enable Audit Data** parameter in the relevant spreadsheet template to **Yes**.

When to Enable Audit for the Enterprise

Setting the **Enable Audit Data** configuration parameters to **Yes** for the enterprise may affect performance. Consider enabling audit in this way only for ongoing integrations.

Purging Audit Data

You may want to purge audit data, typically when disposing of personally identifiable information in worker records. By default, the purging of audit data is disabled for the enterprise and you can’t override this setting at the enterprise level. However, you can include the following SET instruction in individual .dat files:

```
SET PURGE_AUDIT_DATA Y
```

In spreadsheet templates, you can set the **Purge Audit Data** parameter to **Yes**. These instructions purge the audit data for the business objects in the .dat file or spreadsheet.

Restrictions exist on enabling audit and purging data at the same time. You can’t include both of the following SET instructions in a single .dat file:

```
SET ENABLE_AUDIT_DATA Y
SET PURGE_AUDIT_DATA Y
```

Similarly, you can’t include both of the following parameter values in a spreadsheet template:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Audit Data</td>
<td>Yes</td>
</tr>
<tr>
<td>Purge Audit Data</td>
<td>Yes</td>
</tr>
</tbody>
</table>

To purge audit data when auditing is enabled for the enterprise, you must disable audit in the same .dat file or spreadsheet template. That is, you must do one of the following:

- Include a **SET_ENABLE_AUDIT_DATA N** instruction in the same .dat file.
- Set the **Enable Audit Data** parameter in the spreadsheet template to **No**.
Correcting Load Errors for Audited Objects

When audit is enabled for an object, you can’t correct load errors for it in the HCM Data Loader user interface. For example, your .zip file may contain both Worker and Job objects. If you include the `SET ENABLE_AUDIT_DATA Y` instruction in the Worker.dat file only, then you can’t correct load errors from that file in the user interface. However, you can correct errors in the Job objects, as audit isn’t enabled in the Job.dat file.

Related Topics

- Enable Audit for Oracle HCM Cloud Business Objects
- The SET Instruction
- Set Spreadsheet Template Parameters

FAQs for Preparing to Use HCM Data Loader

Can I load data in multiple languages?

Yes. Language packs for the required languages must be installed. For cloud implementations, requested language packs may be installed before the environment is delivered. Alternatively, you can request language packs by raising a service request (SR).
5 Data Preparation

Guidelines for Preparing the Source Data

This topic describes some ways in which you can prepare your data for loading to Oracle HCM Cloud and ensure that it loads successfully.

Reviewing and Cleansing the Source Data

Identify the business objects that you’re planning to upload to Oracle HCM Cloud and their source systems. Review this source data, and verify that it’s both accurate and current. Correct any problems before you attempt to extract it. For example:

- Ensure that a manager is identified for every worker and that the information is accurate.
- For jobs and positions, ensure that correct job codes and titles exist in the source systems.
- For worker history, establish the accuracy of any historical data. Understand whether all historical data must be uploaded or just key events, such as hire, promotion, and termination.

Preparing the source data in this way minimizes the problems that can occur when you upload data to Oracle HCM Cloud. It also makes it less likely that you load inaccurate data to the new environment.

Extracting Data from the Source System

You must define mappings between your source data and the Oracle HCM Cloud business-object model by comparing source and target attributes. To review the structure of a business object in Oracle HCM Cloud:

1. Open the View Business Objects page in the Data Exchange work area.
2. Search for the object by entering its name in the Business Object query-by-example field and pressing Enter.
3. Click the object name to open the Business Object Details page for the object. This page provides comprehensive information about the object’s structure and attributes.

You must also define the transformation logic and build extraction routines. You can use tools that are native to the source system, such as PL/SQL in Oracle E-Business Suite or SQR in Oracle PeopleSoft. Alternatively, you can use an Extract, Transform, and Load tool, such as Oracle Data Integrator or PowerCenter Informatica.

Validating the Source Data Before Upload

HCM Data Loader validates your data during both the import and load phases of data loading. The Data File Validator tool, available from My Oracle Support, can perform most of the data-formatting validations before you attempt to load the data. You run this utility in your source environment to test your generated .dat files. The utility generates a list of validation errors in HTML format. You can correct the errors in the .dat files before you load them.

You can download the Data File Validator tool from the My Oracle Support document, Data File Validator Tool for HCM Data Loader (document ID 2022617.1).

Related Topics

- Data File Validator Tool for HCM Data Loader (2022617.1)
General Data Considerations for HCM Data Loader

This topic describes some general aspects of data preparation. Follow these rules to ensure successful data upload.

Unchanged Attributes

When you update existing data in Oracle HCM Cloud, you supply:

- A unique identifier for the record being updated
- The attributes that have changed

Any attributes that you omit retain their current values. For performance reasons, you’re recommended not to include attributes that haven’t changed.

Note: For date-effective records, all attributes that you supply are updated for the specified date range.

Null Attribute Values

To set an attribute value explicitly to null, you must supply the #NULL token as the attribute value. You can’t simply leave the attribute blank.

Lookup-Validated Attributes

For nonflexfield attributes that are defined in Oracle HCM Cloud as lookup types, you can specify either the lookup code or its meaning. For example, you can specify the sex of a person using either the lookup codes or the lookup meanings shown in this table.

<table>
<thead>
<tr>
<th>Lookup Code</th>
<th>Lookup Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Male</td>
</tr>
<tr>
<td>F</td>
<td>Female</td>
</tr>
</tbody>
</table>

You’re recommended to use the lookup code, as the lookup meaning can be translated. The language of the lookup meaning must match the language of the user who’s uploading the data.

Different rules apply to flexfield attributes that are defined as lookup types.

Numeric Values

For numeric values, only the decimal separator is supported. Don’t include currency symbols, scientific notation, or thousands separators. To set an existing numeric value to null, supply the #NULL token as the attribute value.
Date and Time Attributes

The expected formats for date and time values are shown in this table.

<table>
<thead>
<tr>
<th>Date or Time</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>YYYY/MM/DD</td>
</tr>
<tr>
<td>Time</td>
<td>YYYY/MM/DD HH24:MI:SS</td>
</tr>
</tbody>
</table>

For example: 2016/11/05 14:20:00

To set an existing date or time value to null, supply the #NULL token as the attribute value.

Related Topics
- Guidelines for Loading Flexfield Data

How You Load Images, Attachments, and Large Strings

You can load both Character Large Objects (CLOB) and Binary Large Objects (BLOB) using HCM Data Loader. However, the way in which you provide data for these objects is unique to these attributes. Instead of supplying the data directly in the data (.dat) file, you supply it in a separate file. You supply the name of this file as the value of the relevant attribute in the data file. This example shows a data file for the Document Attachment component. The File attribute references a text attachment file in each MERGE line.

```
METADATA|DocumentAttachment|DocumentType|File|PersonNumber|...
MERGE|DocumentAttachment|Drivers License|file01.txt|23901|...
MERGE|DocumentAttachment|Drivers License|file02.txt|64235|...
```

This approach is used because data for these data types can be very large. Also, data to be loaded directly rather than by attachment may need new-line characters, making it complex to include in the business-object data file.

How to Supply CLOB and BLOB Files

To load data into a CLOB attribute, you supply the data in a separate file. You place the file in a ClobFiles folder in the same .zip file as the business-object data file. Similarly, to load data or upload an attachment to a BLOB attribute, you supply the data or file to attach in a BlobFiles folder. The data type of the attribute that’s used to load a large object or attachment determines which folder to use.
The business-object documentation specifies the data types of all attributes. For example, the **File** attribute of the Document Attachment component has the BLOB data type. Therefore, you place referenced files in the **BlobFiles** folder, as shown in this figure.

File names in the **ClobFiles** and **BlobFiles** folders can include UTF-8 single-byte characters. For example, file names can include the alphanumeric characters a through z, A through Z, and 0 through 9, underscore (_), hyphen (-) and parentheses (). Most file-name extensions are supported, though CLOB data is usually supplied in text (.txt) files.

### How You Supply Source Keys

The source key is made up of two attributes, **SourceSystemOwner** and **SourceSystemId**. If you use a source key to identify a record being merged, then you can also use source keys to identify integration-enabled foreign objects. For example, you can identify the parent record of a child component using a source key. You can’t use source keys for foreign-object references if you aren’t also using a source key for the local record.

This topic describes how to supply source keys for both local records and foreign-object references. It also describes how default source keys are constructed.
Defining Source-System Owner Values

The SourceSystemOwner attribute is common to all source keys supplied in a record. Therefore, all foreign objects that you identify using source keys must have the same SourceSystemOwner value as the record being merged. You must define the SourceSystemOwner value by updating the HRC_SOURCE_SYSTEM_OWNER lookup type before you use source keys.

Specifying Source Keys for Local Records

To identify a local record uniquely using a source key, you supply values for both the SourceSystemId and SourceSystemOwner source-key attributes. This example shows how to identify a job object using a source key.

```
METADATA|Job|SourceSystemId|SourceSystemOwner|JobCode|JobName|SetCode|EffectiveStartDate|EffectiveEndDate
MERGE|Job|12349|EBS-UK|SE|Software Engineer|COMMON|2010/01/01|4712/12/31
```

Specifying Source Keys for Foreign-Object References

To use a source key for a foreign-object reference, you append the hint (SourceSystemId) to the surrogate ID attribute for the foreign object. This example shows how to identify a Job object in an assignment record using a source key. JobId is the surrogate ID attribute for the Job object, to which you append the hint (SourceSystemId) in the METADATA line. You must have created the Job object using HCM Data Loader with the supplied source key.

```
METADATA|Assignment|SourceSystemId|SourceSystemOwner|JobId(SourceSystemId)|EffectiveStartDate|EffectiveEndDate
MERGE|Assignment|234234|EBS-UK|12349|2013/01/01|4712/12/31
```

**Note:** You can use source keys for foreign objects only if those objects are integration-enabled. The integrated business-object documentation identifies which objects are integration enabled.

Default Source Keys

If you specify no source key when you create a record using HCM Data Loader, then a default source key is generated. The default SourceSystemOwner value is FUSION and the SourceSystemId is the surrogate ID. You can use source keys to identify records with default source keys. You can also update existing source keys, including default source keys, using the Source Key object. However, no process exists for extracting source-key information.

**Related Topics**

- Source Keys
- Define the Source-System Owner

How You Update Source Keys

When you load data using HCM Data Loader, you can provide a source key. The source key is a value, usually generated from a legacy environment, that identifies the record uniquely in that environment. You can use the source key to refer to the object both when maintaining it and when referring to it from another object. If you specify no source key, then a default source key is generated. You can update both default and locally defined source keys for integration-enabled objects. This topic describes how to update source keys.
Loading Updates to Source Keys

To update the source key associated with any record, you load a SourceKey.dat file. In the file, you supply both a reference to the record to update and the new source-key value.

This example SourceKey.dat file updates the source key for a person address that's identified by its existing source key. The BusinessObject and Component attributes are used to identify the type of record being updated. For both attribute values, you supply the relevant file discriminator.

```
METADATA|SourceKey|BusinessObject|Component|OldSourceSystemId|OldSourceSystemOwner|NewSourceSystemId|NewSourceSystemOwner
MERGE|SourceKey|Worker|PersonAddress|2342|FUSION|1422-HOME|VISION
```

How You Supply User Keys

User keys are specific to the business object component that you're loading. The integrated business-object documentation identifies the user-key attributes available for all business-object components and foreign-object references.

The user-key attributes are required when you create a record. They're also required when you update it, unless you supply a different key type to identify the record uniquely. This topic describes how to use user keys to identify local records and refer to foreign objects. It also describes how changes to user-key values affect their usefulness as keys.

Specifying User Keys for Local Records

A user key can be made up of multiple attributes. You must supply them all if you're not using another key type, such as a source key, to identify the record. This example shows how to identify a Job object by its user key, which is made up of the JobCode and SetCode attributes.

```
METADATA|Job|JobCode|JobName|SetCode|EffectiveStartDate|EffectiveEndDate
MERGE|Job|SE|Software Engineer|COMMON|2010/01/01|4712/12/31
```

Specifying User Keys for Foreign-Object References

In this example, an Assignment object is uniquely identified by its source key. However, a user key is used for the foreign-object reference to the associated Job object.

```
METADATA|Assignment|SourceSystemId|SourceSystemOwner|JobCode|SetCode|EffectiveStartDate|EffectiveEndDate
MERGE|Assignment|234234|EBS-UK|SE|COMMON|2013/01/01|4712/12/31
```

Managing Changed User-Key Values

The values of some user keys aren’t fixed. For example, you can change the name of an organization or location. As user keys can change, using them for historical references is difficult. If you’re loading date-effective history for a business-object component where the user key has changed, then you must also supply a source key. This approach enables HCM Data Loader to identify related date-effective records correctly to form the object that you’re loading.

Related Topics
- User Keys
Examples of Supplying Oracle Fusion Surrogate IDs

An Oracle Fusion surrogate ID is assigned automatically to a new record when you save it to the database. Surrogate IDs are specific to the business-object component that you’re loading. The integrated business-object documentation identifies the surrogate ID attribute for all business-object components and foreign-object references. This topic describes how to use surrogate IDs to identify local records and provide foreign-object references.

Note: Oracle HCM Cloud customers are unlikely to have access to the Oracle Fusion surrogate ID.

Supplying Surrogate IDs for Local Records

This example shows how to use a surrogate ID to identify a Job component. For this component, the JobId attribute is its surrogate ID.

```
METADATA|Job|JobId|JobName|EffectiveStartDate|EffectiveEndDate
MERGE|Job|13413|Software Engineer - Java|2013/01/01|4712/12/31
```

Supplying Surrogate IDs for Foreign-Object References

This example shows an Assignment component that’s uniquely identified using a source key. The record includes a foreign-object reference to the associated Job object, which is identified by its surrogate ID, JobId.

```
METADATA|Assignment|SourceSystemId|SourceSystemOwner|JobId|EffectiveStartDate|EffectiveEndDate
MERGE|Assignment|234234|EBS-UK|13413|2013/01/01|4712/12/31
```

Related Topics
- Oracle Fusion GUIDs and Surrogate IDs

Examples of Supplying Oracle Fusion GUIDs

The Oracle Fusion GUID (Globally Unique Identifier) is a hexadecimal value that’s assigned automatically to a record when it’s saved to the database. This topic shows how to use GUIDs to identify local records and provide foreign-object references.

Supplying GUIDs for Local Records

When you supply a GUID value to identify the record being merged or deleted, you use the attribute name GUID, regardless of the business-object component. This example shows how to specify a GUID value to identify a Job component.

```
METADATA|Job|GUID|JobName|EffectiveStartDate|EffectiveEndDate
MERGE|Job|2342UJFH2323|Software Engineer - Java|2013/01/01|4712/12/31
```
Supplying GUIDs for Foreign-Object References

To use a GUID for a foreign-object reference, you append the hint (GUID) to the surrogate ID attribute for the object to which you're referring. This example shows an Assignment component that's identified using source keys. The record includes a foreign-object reference to the associated Job object, which is identified by its GUID. JobId is the surrogate ID attribute for the Job object.

```
METADATA|Assignment|SourceSystemId|SourceSystemOwner|JobId(GUID)|EffectiveStartDate|EffectiveEndDate
MERGE|Assignment|234234|EBS-UK|2342UJFJ323|2013/01/01|4712/12/31
```

You can use GUIDs for foreign objects only if they're integration enabled. The integrated business-object documentation identifies which foreign objects are integration enabled.

Related Topics

- Oracle Fusion GUIDs and Surrogate IDs

How You Manage Reserved Characters

A few characters are reserved in the context of an HCM Data Loader data file. That is, they have a specific meaning by default, and you can't include them in attribute values unless you identify them explicitly as data. This topic explains how to use the reserved characters in data that you're uploading. It also explains how to override the default reserved characters.

Using Reserved Characters in Attribute Values

By default, these characters are reserved:

- Delimiter (vertical bar |)
- Newline character (n)
- Escape (backslash \)

To include the newline (n) and vertical bar (|) characters in attribute values, you precede them immediately with the escape character (backslash \). For example:

```
METADATA|Address|AddressLine1
MERGE|Address|The Stables\|Main Allan
```

This entry enables the vertical bar to appear in the attribute value for address line 1:

The Stables|Main Allan

To include the newline character in a value, you specify \n. For example:

```
METADATA|Address|AddressLine1
MERGE|Address|The Stables\nMain Allan
```

This entry results in the following value for address line 1:

The Stables
Main Allan
Overriding the Reserved Characters

You can override the reserved characters for a file using the SET file-line instruction, which must appear before any METADATA lines in the file. The SET commands for overriding reserved characters have this format:

```
SET FILE_ESCAPE <new_value>
SET FILE_DELIMITER <new_value>
SET FILE_NEW_LINE <new_value>
```

The new value can be up to 10 characters. For example, you could set the newline character to newline and the file delimiter to comma (,) using the following SET commands:

```
SET FILE_DELIMITER ,
SET FILE_NEW_LINE newline
```

In this case, METADATA and MERGE lines could appear as follows:

```
METADATA,Address,AddressLine1
MERGE,Address,TheSteading\newlineKier Allan
```

💡 Tip: You can also override the reserved characters for the enterprise by setting relevant configuration parameters. For example, set the **File Delimiter** configuration parameter to override the default file delimiter for the enterprise.

**Related Topics**
- The SET Instruction

How You Load and Maintain Translated Objects

In environments where multiple languages are enabled, you can use HCM Data Loader to upload translated objects. You specify the character set of the data file by naming any Java-supported character set on the **File Character Set** configuration parameter. The default character set is UTF-8.

### Loading Translated Objects

Loading translated objects is a two-stage process:

1. You create the object by loading the base-language version. At this stage, translation records are created for all enabled languages, but they hold base-language versions of any translatable values. For example, if US English is your base language, then translation records hold US English versions of translatable values.
2. You load the translated values as corrections to the base-language object. To perform this step, you use data-file templates that are provided specifically for translation. One translation data-file template is provided for each business-object component that includes translatable values.

For example, you may create the Sales Manager job in an environment where US English is the base language. If French, German, and Spanish are also enabled, then the object is created as shown in this table.

<table>
<thead>
<tr>
<th>Language</th>
<th>Source Language</th>
<th>Job Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>US English</td>
<td>US English</td>
<td>Sales Manager</td>
</tr>
</tbody>
</table>
Once this object exists, you can load a single translation data file (JobTranslation.dat) to correct the French, German, and Spanish versions of the job name. Alternatively, you can load a translation file for each language if you prefer. You can deliver translation files either in the same .zip file as the original object or separately. However, you can’t deliver them before the base-language object exists.

### Updating Translated Data

To update existing translated data, you can upload just the relevant translation file, provided that the object isn’t date effective.

You must also update the base-language object if:

- The object is date effective.
- The object currently has no date-effective record with the same effective start date as the new translation value.

This requirement exists to ensure that the effective dates of the base-language and translation objects remain the same.

You can’t delete translation objects in isolation. Translation objects are deleted automatically when you delete the associated object. For example, when you delete a Job Family object any associated translation objects are deleted automatically.

### Translation-File Discriminators

Unique file discriminators exist for the translation files and are identified in relevant files. For example, the file discriminator for the file JobTranslation.dat is JobTranslation.

This example shows a Job.dat file followed by the associated JobTranslation.dat file.

```plaintext
METADATA|Job|SourceSystemOwner|SourceSystemId|EffectiveStartDate|EffectiveEndDate|JobCode|Name|ActiveStatus
MERGE|Job|EBS-UK|JB2ACC44|2010/01/01|2014/04/04|ACADM|Accounts Administrator|A
MERGE|Job|EBS-UK|JB2ACC44|2014/04/05|4712/12/31|ACADM|Accounts Clerk|A

METADATA|JobTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|EffectiveEndDate|Language|Name
MERGE|JobTranslation|EBS-UK|JB2ACC44|2010/01/01|2014/04/04|ES|Administrador de Cuentas
MERGE|JobTranslation|EBS-UK|JB2ACC44|2014/04/05|4712/12/31|ES|Cuentas Clerk
```

Related Topics

- HCM Data Loader Configuration Parameters

### How You Include Source-System References in Data Files

You can include source-system references in each data line in a file. They enable you to record the database-table name, column names, and attribute values from the source system. You can see these values on the Object Errors page for an
object that fails to load. Therefore, you can easily identify the data source. Source-system references are optional. This topic
describes how to construct source-system references, which are made up of a name and a value.

Source-System Names

You specify source-system database-table and column names in the relevant METADATA line. To specify the source-system
database-table name, you add the following entry to the METADATA line:

```
SourceRefTableName=<table name>
```

You can specify up to 10 source-system column names in the same METADATA line using the SourceRef001 to
SourceRef010 tags. For example:

```
METADATA|Job|SourceRefTableName=PER_JOBS|SourceRef001=JOB_ID|SourceRef002=EFFECTIVE_START_DATE|
        |    |SourceRef003=EFFECTIVE_END_DATE
```

Source-system references can appear anywhere in the METADATA line after the instruction and discriminator values.

Source-System Values

Supply the source-system values on each data line, ensuring that they appear in the order specified on the METADATA line.
In data lines, you must leave the source-system database-table name blank. This value appears in the METADATA definition
only. For example:

```
METADATA|Job|SourceRefTableName=PER_JOBS|SourceRef001=JOB_ID|SourceRef002=EFFECTIVE_START_DATE|
        |    |SourceRef003=EFFECTIVE_END_DATE
MERGE|Job||135|2010/01/01|4712/12/31
MERGE|Job||136|2010/01/01|4712/12/31
```

Related Topics

- How You Review and Correct Errors

Data Deletion Using HCM Data Loader

You can delete many objects using HCM Data Loader, regardless of whether they were loaded using HCM Data Loader. This
topic explains how to delete an object or its components and identifies any restrictions.

⚠️ Tip: As you can’t recover a deleted object, try to correct your data instead of deleting and recreating it.

What You Can and Can't Delete

You can delete:

- Most complete business objects, except Worker.
- Most individual business-object components.
- Some child components of the Worker object. For example, you can delete the Person Email component of a worker
  object.
When you delete a parent object, its child components and any translation objects are also deleted. For example, to delete a Grade object and its child components, you create a DELETE instruction for the Grade discriminator. To delete only a Grade Rate Value child component, you create a DELETE instruction for the GradeRateValue discriminator.

You can’t delete:

- Individual date-effective records
- Individual translation objects
- Worker objects

>Note: Before you delete an object, ensure that other business objects don’t refer to it.

For information about whether you can delete a component, see the business-object details on the View Business Objects page in the Data Exchange work area.

DELETE Instructions

To delete an object, include the DELETE instruction in the relevant data file. For example, you could delete a Job Family object by including these lines in the JobFamily.dat file:

```
METADATA|JobFamily|EffectiveStartDate|EffectiveEndDate|JobFamilyName
DELETE|JobFamily|2012/10/01|4712/12/31|Sales01
```

These rules apply:

- You can’t include DELETE instructions in translation data files.
- You must not supply a DELETE instruction for a record that has a MERGE instruction in the same file. HCM Data Loader doesn’t know which instruction to process first.

Deleting Date-Effective Objects

To delete a date-effective object identified by user keys, you must supply both the effective start and effective end dates. If the object is identified using any of the other key types, then the effective start and end dates are optional.
Loading Date-Effective Data

Overview of Loading Date-Effective Objects

Many Oracle HCM Cloud objects are date-effective. That is, they retain a history of changes, each of which has effective start and end dates. Professional users can retrieve and edit the version of an object as of a specified date. You can load date-effective objects and their date-effective history using HCM Data Loader. This topic provides some general rules about loading date-effective objects.

Attributes of Date-Effective Records

This table introduces the attributes that occur in date-effective records.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EffectiveStartDate</td>
<td>The start date for the attribute values. This value is required for all date-effective records.</td>
</tr>
<tr>
<td>EffectiveEndDate</td>
<td>The end date for the attribute values. If you leave this value blank, then the date-effective record continues by default to the end of time.</td>
</tr>
<tr>
<td>EffectiveSequence</td>
<td>When multiple changes per day occur, this attribute identifies their order.</td>
</tr>
<tr>
<td>EffectiveLatestChange</td>
<td>For objects with multiple changes per day, this attribute identifies the latest record for the effective start date.</td>
</tr>
</tbody>
</table>

Supplying Date-Effective History

You can decide how much history to load for new objects, but the history that you provide must be complete and valid. Although you can supply date-effective records in any order in the .dat file, no break in the dates can occur. When you create a date-effective object, only the first date-effective record is processed as a new record. Later-dated records are updates to the first record. To retain values from the previous date-effective record, you can either leave those attribute values blank or repeat the value that’s to continue.

Note: Leaving an attribute value blank doesn't set the attribute to null. To change a value to null, you must supply the #NULL token as the attribute value.
Key Values

HCM Data Loader groups records into logical objects, where a logical object is one occurrence of the business object, such as a Worker. The records belonging to a logical object are identified by the unique key for the component. Therefore, the key value must be the same throughout the date-effective history. You can use any of the four key types.

Creating Date-Effective Objects

When you create business objects containing date-effective components using HCM Data Loader:

- Include the `SET PURGE_FUTURE_CHANGES Y` instruction at the start of the data file. This instruction sets the HCM Data Loader date-effective maintenance mode to Replace. In this mode, which is the default, the contents of the data file replace any existing data.
- If you’re including object history, then supply the #NULL token for all attributes that must have null values.

Updating Date-Effective Objects

When you load updates to objects that contain date-effective components:

- Include the `SET PURGE_FUTURE_CHANGES N` instruction at the start of the data file. This instruction sets the HCM Data Loader date-effective maintenance mode to Retain. In this mode, existing future-dated changes are retained. If you don’t include this instruction, then any existing future-dated changes for the date range of your updates are purged.
- Always supply the effective start date of each change. The effective end date is optional. However, you must set the effective end date to #RETAIN if you don’t want to correct any future-dated records. Leave the effective end date null if you want a change to apply until the end of time.

> Note: When you update date-effective objects, the values that you supply are applied to every record in the specified date-range. However, some attributes, such as ActionCode, are protected from update.

How You Load Objects with Multiple Changes per Day

For most date-effective objects, only one record exists for a specific date. Therefore, the object history shows only the last update to each attribute on any day. However, for some objects, such as Assignment, a record exists for each update in a day. Such objects include an effective sequence number to show the update order. Changes are cumulative, and the record with the highest effective sequence number is available to transactions for that date. This topic describes how to load and maintain objects that support multiple changes per day.

Effective Sequence Number

When you load date-effective history for an object with multiple changes on a single date, the EffectiveSequence value must start at 1 and increase sequentially. You can’t repeat an EffectiveSequence value for the same logical
object on the same date, nor can gaps exist in the sequence. If only one change exists for an effective start date, then EffectiveSequence must be 1.

Note: Don’t leave EffectiveSequence blank when you’re loading multiple changes for a single day. Without this information, the order in which records with the same effective start date are processed is undefined.

Effective Latest Change

When you report multiple changes on the same effective start date, the EffectiveLatestChange attribute of the last record in the sequence must be Y. For all earlier records, the attribute must be N. EffectiveLatestChange is a required attribute for records that support multiple changes per day. If only one change exists on an effective start date, then EffectiveLatestChange must be Y.

Effective Start and End Dates

The effective end date of any record with EffectiveLatestChange set to N must be the same as its effective start date.

Loading Date-Effective History with Multiple Changes Per Day: Example

This example loads date-effective history for an Assignment object.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>Assignment</th>
<th>SourceSystemId</th>
<th>EffectiveStartDate</th>
<th>EffectiveSequence</th>
<th>EffectiveLatestChange</th>
<th>EffectiveEndDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>Assignment</td>
<td>2724</td>
<td>2010/06/08</td>
<td>1</td>
<td>Y</td>
<td>2012/03/03</td>
</tr>
<tr>
<td>MERGE</td>
<td>Assignment</td>
<td>2724</td>
<td>2012/03/04</td>
<td>1</td>
<td>N</td>
<td>2012/03/04</td>
</tr>
<tr>
<td>MERGE</td>
<td>Assignment</td>
<td>2724</td>
<td>2012/03/04</td>
<td>2</td>
<td>N</td>
<td>2012/03/04</td>
</tr>
<tr>
<td>MERGE</td>
<td>Assignment</td>
<td>2724</td>
<td>2012/03/04</td>
<td>3</td>
<td>Y</td>
<td>2012/06/01</td>
</tr>
<tr>
<td>MERGE</td>
<td>Assignment</td>
<td>2724</td>
<td>2012/06/02</td>
<td>1</td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>

This example observes these rules:

- No gaps exist in the dates.
- The unique key is identical for all date-effective records.
- EffectiveSequence starts at 1 for all new effective start dates and is sequential for matching effective start dates.
- EffectiveLatestChange is Y when the record is the last change on an effective start date. Otherwise, EffectiveLatestChange is N.
- The EffectiveEndDate matches the EffectiveStartDate when EffectiveLatestChange is N.

Updating Objects with Multiple Changes Per Day

When updating an existing object with multiple changes per day, you may not know the next available sequence number. In this case, leave the EffectiveSequence attribute blank. The next number in the existing sequence is generated automatically.

To correct an existing record, you must supply the existing values for these attributes to identify the record correctly:

- Effective Start Date
- Effective End Date
- Effective Sequence
- Effective Latest Change
Options for Updating Date-Effective Objects with Future-Dated Records

When you update a date-effective object, your changes may affect more than one of its existing date-effective records. Also, some of the affected records may have effective start dates that are later than the effective start date of the update. Such records are known as future-dated records. This topic describes how updates to future-dated records are processed in each of the maintenance modes. It also shows how to set the maintenance mode.

This table shows the date-effective history of a Job object.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective End Date</th>
<th>Job Code</th>
<th>Job Name</th>
<th>Regular Temporary</th>
<th>Full Part Time</th>
<th>Active Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>2012/01/09</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>T</td>
<td>PART_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2012/01/10</td>
<td>2012/03/03</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>T</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>4712/12/31</td>
<td>ACC1</td>
<td>Accounts Administrator</td>
<td>T</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
</tbody>
</table>

If you update the object specifying an effective start date of 2011/01/01, then a new date-effective record is generated for 2011/01/01. This update changes the effective end date of the 2010/06/08 record. It also affects both the 2012/01/10 and 2012/03/04 records. The exact effect on these existing future-dated records depends on:

- The maintenance mode, which can be either Retain or Replace
- The effective end date value, if any, that you supply

Retain Mode or Replace Mode

This table summarizes the combined effects of the maintenance mode and effective end date on existing future-dated records.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Effective End Date Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retain</td>
<td>#RETAIN</td>
<td>Existing attribute values in future-dated records are retained. Your updates don’t affect these values. All existing records are retained.</td>
</tr>
<tr>
<td>Retain</td>
<td>#ALL</td>
<td>Changed attribute values are carried forward to all future-dated records until the end of time. All existing records are retained.</td>
</tr>
</tbody>
</table>
Loading Date-Effective Data

<table>
<thead>
<tr>
<th>Mode</th>
<th>Effective End Date Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retain</td>
<td>Specific date other than 4712/12/31</td>
<td>Changed attribute values are carried forward to future-dated records until the specified end date. All existing records are retained.</td>
</tr>
<tr>
<td>Replace</td>
<td>Specific date or blank</td>
<td>A single record replaces existing date-effective records for the date range specified on the update.</td>
</tr>
</tbody>
</table>

**Tip:** Leaving the effective end date blank or setting it to 4712/12/31 has the same effect as specifying the #ALL token. However, you must use the #ALL token to carry forward changes for objects that have a specific end date.

Retain mode is recommended when you’re applying an incremental update to an existing record. Replace mode, which is the default mode, is useful during data migration, when you’re uploading the complete data for a record.

**Note:** Replace mode may have unexpected consequences if you upload less than the complete data for a record.

In both modes:
- A new date-effective record is generated for the object if you specify an effective start or end date for which no date-effective record currently exists.
- Only the attributes for which you supply values are updated.

Setting the Maintenance Mode

Use the SET command to set the maintenance mode for a date-effective object. This table shows the set command to include in the data file.

<table>
<thead>
<tr>
<th>Mode</th>
<th>SET Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retain</td>
<td>SET PURGE_ FUTURE_CHANGES N</td>
</tr>
<tr>
<td>Replace</td>
<td>SET PURGE_ FUTURE_CHANGES Y</td>
</tr>
</tbody>
</table>

If you include neither of these SET commands in a data file, then the default mode is Replace. The default mode can be overridden for the enterprise using the **Date-Effective Update Mode** configuration parameter.
Example of Retaining Attribute Values in Future-Dated Records

When you update a *date-effective object*, you can leave attribute values in future-dated records unchanged. To retain these values, you set the maintenance mode to Retain and specify the #RETAIN token instead of an *effective end date*. This token ensures that your changes take effect from the supplied *effective start date* to the start of the next record. If no future-dated records exist, then your changes apply until the end date of the record. This topic provides an example showing the effects of Retain mode.

Retaining Future-Dated Attribute Values in an Assignment Object

Consider the date-effective history of the Assignment object shown in this table.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective Sequence</th>
<th>Effective End Date</th>
<th>Action Code</th>
<th>Job Code</th>
<th>Grade</th>
<th>Location</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>1</td>
<td>2012/03/03</td>
<td>HIRE</td>
<td>ACC1</td>
<td>IC2</td>
<td>HQ</td>
<td>40</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>1</td>
<td>2012/03/04</td>
<td>PROMOTION</td>
<td>ACC1</td>
<td>IC4</td>
<td>HQ</td>
<td>40</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>2</td>
<td>2012/03/04</td>
<td>TRANSFER</td>
<td>ACC1</td>
<td>IC4</td>
<td>LVP</td>
<td>40</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>3</td>
<td>2012/06/01</td>
<td>JOB_CHANGE</td>
<td>ACC3</td>
<td>IC4</td>
<td>LVP</td>
<td>40</td>
</tr>
<tr>
<td>2012/06/02</td>
<td>1</td>
<td>4712/12/31</td>
<td>JOB_CHANGE</td>
<td>ACC2</td>
<td>IC4</td>
<td>LVP</td>
<td>40</td>
</tr>
</tbody>
</table>

To change the working hours to 37.5 from 2012/01/10 without affecting future-dated records, you use this data file:

```
SET PURGE_FUTURE_CHANGES N
METADATA|Assignment|SourceSystemId|EffectiveStartDate|EffectiveSequence|EffectiveLatestChange|
|EffectiveEndDate|ActionCode|NormalHours
MERGE|Assignment|2724|2012/01/10||Y|#RETAIN|ASG_CHANGE|37.5
```

After update, the Assignment object is as shown in this table.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective Sequence</th>
<th>Effective End Date</th>
<th>Action Code</th>
<th>Job Code</th>
<th>Grade</th>
<th>Location</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>1</td>
<td>2012/01/09</td>
<td>HIRE</td>
<td>ACC1</td>
<td>IC2</td>
<td>HQ</td>
<td>40</td>
</tr>
<tr>
<td>2012/01/10</td>
<td>1</td>
<td>2012/03/03</td>
<td>ASG_CHANGE</td>
<td>ACC1</td>
<td>IC2</td>
<td>HQ</td>
<td>37.5</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>1</td>
<td>2012/03/04</td>
<td>PROMOTION</td>
<td>ACC1</td>
<td>IC4</td>
<td>HQ</td>
<td>40</td>
</tr>
</tbody>
</table>
Attributes in records with effective start dates after 2012/01/10 retain their current values.

Note: As recommended, the data file includes no effective sequence number. Therefore, the next available number in the sequence is allocated automatically.

Options for Carrying Forward Changed Attribute Values in Retain Mode

When you update a date-effective object in Retain mode, you may want to carry changes forward to future-dated records. In this case, you can either specify an effective end date or carry the changes forward to all existing future-dated records. This topic describes how to achieve each outcome and what to consider when deciding how to proceed.

Specifying an Effective End Date

This table shows the date-effective history of a Job object.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective End Date</th>
<th>Job Code</th>
<th>Job Name</th>
<th>Regular Temporary</th>
<th>Full Part Time</th>
<th>Active Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>2012/01/09</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>T</td>
<td>PART_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2012/01/10</td>
<td>2012/03/03</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>T</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>4712/12/31</td>
<td>ACC1</td>
<td>Accounts Administrator</td>
<td>T</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
</tbody>
</table>

To change the Regular Temporary value from T to R for the period from 2011/03/04 to 2014/04/04, you include these instructions in the .dat file:

```
SET PURGE_FUTURE_CHANGES N
METADATA|Job|SourceSystemId|EffectiveStartDate|EffectiveEndDate|RegularTemporary
MERGE|Job|45346|2011/03/04|2014/04/04|R
```

After update, the Job object is as shown in this table.
A new date-effective record is created for both the start and end dates of the change. The Regular Temporary attribute is updated for the specified period. All other attributes keep their existing values.

### Applying Changes to All Future-Dated Records

You can use the #ALL token in place of an effective end date. This token specifies explicitly that your changes apply to all future-dated records, regardless of the object’s end date.

For example, you want to make these changes to the Job object:

- Change the Regular Temporary value from T to R.
- Set the effective start date of the change to 2011/03/05.
- Apply the changes to all future-dated records.

You include these instructions in the data file:

```
SET PURGE_FUTURE_CHANGES N
METADATA|Job|SourceSystemId|EffectiveStartDate|EffectiveEndDate|RegularTemporary
MERGE|Job|45346|2011/03/04|#ALL|R
```

After update, the Job object is as shown in this table.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective End Date</th>
<th>Job Code</th>
<th>Job Name</th>
<th>Regular Temporary</th>
<th>Full Part Time</th>
<th>Active Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>2011/03/03</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>T</td>
<td>PART_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2011/03/04</td>
<td>2012/01/09</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>R</td>
<td>PART_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2012/01/10</td>
<td>2012/03/03</td>
<td>ACC1</td>
<td>Accounts Clerk</td>
<td>R</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>2014/04/04</td>
<td>ACC1</td>
<td>Accounts Administrator</td>
<td>R</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
<tr>
<td>2014/04/05</td>
<td>4712/12/31</td>
<td>ACC1</td>
<td>Accounts Administrator</td>
<td>T</td>
<td>FULL_TIME</td>
<td>A</td>
</tr>
</tbody>
</table>

You can achieve the same result by leaving the effective end date in the .dat file blank or setting it explicitly to 4712/12/31. However, you must use the #ALL tag for any object that has an explicit end-date.
Example of Replacing Attribute Values in Future-Dated Records

When you update a date-effective object in Replace mode, you replace all existing, future-dated records with the uploaded data. You’re recommended to use Replace mode only when the contents of the .dat file must replace the existing data rather than update it.

Replace mode is the default maintenance mode. However, you can use this SET command to select Replace mode explicitly and record the setting in the .dat file:

```
SET PURGE_FUTURE_CHANGES Y
```

Updating Only Selected Attributes of an Object in Replace Mode

This table shows the date-effective history of an Assignment object.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective Sequence</th>
<th>Effective End Date</th>
<th>Action Code</th>
<th>Job</th>
<th>Grade</th>
<th>Location</th>
<th>Normal Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>1</td>
<td>2012/03/03</td>
<td>HIRE</td>
<td>ACC1</td>
<td>IC2</td>
<td>HQ</td>
<td>40</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>1</td>
<td>2012/03/04</td>
<td>PROMOTION</td>
<td>ACC1</td>
<td>IC4</td>
<td>HQ</td>
<td>40</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>2</td>
<td>2012/03/04</td>
<td>TRANSFER</td>
<td>ACC1</td>
<td>IC4</td>
<td>LVP</td>
<td>40</td>
</tr>
<tr>
<td>2012/03/04</td>
<td>3</td>
<td>2012/06/01</td>
<td>JOB_CHANGE</td>
<td>ACC3</td>
<td>IC4</td>
<td>LVP</td>
<td>40</td>
</tr>
<tr>
<td>2012/06/02</td>
<td>1</td>
<td>4712/12/31</td>
<td>JOB_CHANGE</td>
<td>ACC2</td>
<td>IC4</td>
<td>LVP</td>
<td>37.5</td>
</tr>
</tbody>
</table>

To update the working hours to 37.5 from 2012/01/10 in Replace mode, you create the following data file:

```
SET PURGE_FUTURE_CHANGES Y
METADATA|Assignment|SourceSystemId|EffectiveStartDate|EffectiveSequence|EffectiveLatestChange|
|--------|------------|-----------------|------------------|--------------------|---------------------|
MERGE|Assignment|2724|2012/01/10||Y||ASG_CHANGE|37.5
```

After update, the Assignment object is as shown in this table.

<table>
<thead>
<tr>
<th>Effective Start Date</th>
<th>Effective Sequence</th>
<th>Effective End Date</th>
<th>Action Code</th>
<th>Job</th>
<th>Grade</th>
<th>Location</th>
<th>Normal Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010/06/08</td>
<td>1</td>
<td>2012/01/09</td>
<td>HIRE</td>
<td>ACC1</td>
<td>IC2</td>
<td>HQ</td>
<td>40</td>
</tr>
<tr>
<td>2012/01/10</td>
<td>1</td>
<td>4712/12/31</td>
<td>ASG_CHANGE</td>
<td>ACC1</td>
<td>IC2</td>
<td>HQ</td>
<td>37.5</td>
</tr>
</tbody>
</table>
In Replace mode, records with effective start dates that are later than the effective start date of the update are purged. Attribute values that predate the change and for which you supply no values in the data file appear in the new record. This update isn’t reversible.

⚠️ Caution: You’re recommended to supply values for all attributes that currently exist on the object when updating in Replace mode. In this example, you could have provided values for job, grade, and location. As you didn’t supply those values, the values that are current on the effective start date of the update remain in effect.

Examples of Changing First Effective Start Date and Last Effective End Date

For some objects, you can change the first effective start date or last effective end date. This topic provides examples showing how to make these changes.

To change an object’s first effective start date, you include the ReplaceFirstEffectiveStartDate attribute in the .dat file and set it to Y. To change an object’s last effective end date, you include the ReplaceLastEffectiveEndDate attribute in the .dat file and set it to Y. You supply the revised dates on the EffectiveStartDate or EffectiveEndDate attribute, as appropriate. You can make these changes at the same time as other changes. For example, you can change both the start date of an object and its name.

📝 Note: The ReplaceFirstEffectiveStartDate and ReplaceLastEffectiveEndDate parameters don’t appear in object template files or on the Attributes tab of the Business Object Details page. However, you can add them to .dat files for date-effective objects that support these changes.

These restrictions apply to changing first effective start date and last effective end date:

- For some objects, the last effective end date must be 4712/12/31. Any attempt to change this date causes an error.
- You can’t change the first effective start date of the Worker object in this way. Instead, you must use the NewStartDate attribute.
- When you include the ReplaceLastEffectiveEndDate attribute in a .dat file, you must specify a date value for the EffectiveEndDate attribute. You must not supply a #RETAIN, #ALL, or #NULL token in place of the effective end date.

💡 Tip: On the Component Details tab of the Business Object Details page, identify whether you can change a component’s first effective start or last effective end date. For example, in the Date Effectivity section on the Component Details tab for the Job object, the First start date option is selected. Therefore, you can include the ReplaceFirstEffectiveStartDate parameter in the .dat file.

Changing the First Effective Start Date of a Job Object

This example shows how to include the ReplaceFirstEffectiveStartDate attribute in a .dat file.

```
METADATA|Job|JobId|EffectiveStartDate|EffectiveEndDate|ReplaceFirstEffectiveStartDate
MERGE|Job|23452|1950/01/01|4712/12/31|Y
```
Changing the Last Effective End Date of a Recurring Element Entry

This example shows how to include the `ReplaceLastEffectiveEndDate` attribute in a .dat file.

```
METADATA|ElementEntry|ElementEntryId|EffectiveStartDate|EffectiveEndDate|ReplaceLastEffectiveEndDate
MERGE|ElementEntry|4634|2014/01/01|2014/04/05|Y
```
7 Loading Flexfield Data

Guidelines for Loading Flexfield Data

Using HCM Data Loader, you can load data for both descriptive flexfields and extensible flexfields. This topic explains concepts that are common to both types of flexfields.

Configuring Flexfields

Before you generate template files for objects with flexfields, you must configure the flexfields in Oracle HCM Cloud. After the flexfields are configured, the METADATA lines in generated template files include all attributes that are required to load the flexfield data. You also see a Flexfield Attributes tab for relevant object components on the Business Object Details page in the Data Exchange work area.

Flexfield Codes

When loading flexfield data, you must supply the flexfield code in the METADATA line in this format:

FLEX:<flexfield code>

For example, to load data for the job descriptive flexfield, the flexfield code is:

FLEX:PER_JOBS_DFF

Relevant flexfield codes appear in the generated template file and on the Flexfield Attributes tab for the business object.

Flexfield Attributes

Flexfield attribute names are those that you specify when configuring the flexfield. Both descriptive and extensible flexfields have one or more contexts. When you include a flexfield attribute name on the METADATA line for an object, you must also identify the context. You provide a hint value in the following format to identify both the flexfield and the context to which the attribute belongs:

<flexfield attribute name>(<flexfield code>=<context code>)

For example, for the _MAIN_CONTRACT attribute of the PER_CONTRACT_LEG_DDF descriptive flexfield with the CN context code, the entry on the METADATA line is:

.MAIN_CONTRACT(PER_CONTRACT_LEG_DDF=CN)

Business Objects with Multiple Descriptive Flexfields

Some business object components support multiple descriptive flexfields. You can include all descriptive flexfield attributes for every flexfield and configured context on a single METADATA line. This approach is possible because you supply both the flexfield code and the context for each attribute.
Lookup-Validated Flexfield Values

For flexfield values that are validated by lookups, you can supply either the lookup code or the lookup value. To supply the value, you must append the suffix _Display to the attribute name in the METADATA line.

For example, for a configured gender flexfield attribute, you can supply either the lookup code (M or F) or the value (Male or Female). To supply the value, use the attribute name gender_Display, as shown in this example:

```plaintext
METADATA|Job|FLEX:PER_JOBS_DFF|gender_Display(PER_JOBS_DFF=JOBCONTEXT1)|...
MERGE|Job|JOBCONTEXT1|Male|
MERGE|Job|JOBCONTEXT1|Female|
```

Secured Value Sets

A value set is a group of values that validate the content of a flexfield segment. If the value set is security enabled, then it’s available to specific users or processes only. You must ensure that the predefined FUSION_APPS_HCM_ESS_LOADER_APPID user has access to any security-enabled value sets that your flexfields reference. Otherwise, the load fails. For more information about enabling access to secured value sets, see the Securing HCM guide.

Related Topics

- Generate and Use Template Files
- Enable Security for Value Sets
- Secure Access to Value Sets

How You Load Data for Descriptive Flexfields

This topic describes how to construct METADATA and MERGE lines in a .dat file to supply descriptive flexfield data.

Constructing the METADATA Line

Descriptive flexfields extend a business object. You can supply the attributes of the descriptive flexfield on the same MERGE line as the core attributes of the business-object component. On the METADATA line, you must include both the flexfield code and the descriptive flexfield attributes for which you want to load data. In a generated business-object template, the METADATA line automatically includes the flexfield code and attributes for configured descriptive flexfields.

This example shows the METADATA line for the Job business object with entries for the PER_JOBS_DFF descriptive flexfield:

```plaintext
METADATA|Job|FLEX:PER_JOBS_DFF|JobCode|ActiveStatus|FullPartTime|MedicalCheckupRequired|RegularTemporary|EffectiveStartDate|EffectiveEndDate|Name|SetCode|JobFamilyName|_JOB_LEVEL(PER_JOBS_DFF=US)
```

The descriptive flexfield attributes can appear anywhere on the line. You don’t have to append them at the end.

Constructing the MERGE Lines

For each descriptive flexfield for a business-object component, a single MERGE record can have only one context. You specify the context value on the MERGE line against the flexfield code for the descriptive flexfield in the METADATA line.

For example:
On the MERGE line, the context value (US) appears in the same position as the flexfield code (FLEX:PER_JOBS_DFF) on the METADATA line.

Supplying MERGE Lines for Multiple Flexfield Codes and Contexts

When a single business object component supports multiple descriptive flexfields, you can load all of the flexfield data at once. The following example shows METADATA and MERGE lines for the Contract component of the Worker object. Each MERGE line is for a single context.

Updating Global Segments

You can configure both global and context-specific segments for a flexfield. When updating global segments in records that have context-specific segments, set the context value to the existing context value, not Global Data Elements. If you set the context value to Global Data Elements, then existing context-specific values are set to null because the context has changed. For example, consider these two segments of a flexfield record:

<table>
<thead>
<tr>
<th>Segment</th>
<th>Global</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment 1</td>
<td>Yes</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Segment 2</td>
<td>No</td>
<td>CN</td>
</tr>
</tbody>
</table>

If you update segment 1, which is a global segment, then you must set the context to CN, not Global Data Elements. Otherwise, the existing contents of segment 2 are set to null.

Example of Loading Data for Descriptive Flexfields

This topic shows how to construct the METADATA line for the Contract component of the Worker object with multiple configured descriptive flexfields. The METADATA line appears automatically in the generated worker template.

The contract component supports these descriptive flexfields:

- PER_CONTRACT_DF
- PER_CONTRACT_LEG_DDF
Constructing the METADATA Line for the Worker Contract Component

The PER_CONTRACT_DF descriptive flexfield is configured in Oracle HCM Cloud with the context and attribute values shown in this table. The **Hint** column shows the resulting hint for each attribute.

<table>
<thead>
<tr>
<th>Context</th>
<th>Attribute</th>
<th>Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>CONTRACT_GLB</td>
<td>(PER_CONTRACT_DF=Global Data Elements)</td>
</tr>
<tr>
<td>CONTRACT_DF</td>
<td>Currency</td>
<td>(PER_CONTRACT_DF=CONTRACT_DF)</td>
</tr>
</tbody>
</table>

The PER_CONTRACT_LEG_DDF descriptive flexfield is configured in Oracle HCM Cloud with the context and attribute values shown in this table. The **Hint** column shows the resulting hint for each attribute.

<table>
<thead>
<tr>
<th>Context</th>
<th>Attribute</th>
<th>Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH</td>
<td>_MAIN_CONTRACT</td>
<td>(PER_CONTRACT_LEG_DDF=CH)</td>
</tr>
<tr>
<td>CN</td>
<td>_CONST_PROB_DATE</td>
<td>(PER_CONTRACT_LEG_DDF=CN)</td>
</tr>
<tr>
<td>CN</td>
<td>_NDA</td>
<td>(PER_CONTRACT_LEG_DDF=CN)</td>
</tr>
<tr>
<td>CN</td>
<td>_COMPETITION_CLAUSE</td>
<td>(PER_CONTRACT_LEG_DDF=CN)</td>
</tr>
<tr>
<td>CN</td>
<td>_NOTICE_DURATION_UNIT</td>
<td>(PER_CONTRACT_LEG_DDF=CN)</td>
</tr>
</tbody>
</table>

In the generated Worker.dat template file, the Contract METADATA line includes these attribute definitions for the flexfield attributes:

```plaintext
METADATA|Contract|...|CONTRACT_GLB(PER_CONTRACT_DF=Global Data Elements)|
   |    | |_MAIN_CONTRACT(PER_CONTRACT_LEG_DDF=CH)|
   |    | |_CONST_PROB_DATE(PER_CONTRACT_LEG_DDF=CN)|
   |    | |_NDA(PER_CONTRACT_LEG_DDF=CN)|
   |    | |_COMPETITION_CLAUSE(PER_CONTRACT_LEG_DDF=CN)|
   |    | |_NOTICE_DURATION_UNIT(PER_CONTRACT_LEG_DDF=CN)
```

As each attribute has a hint that identifies both the descriptive flexfield and the context, you can define attributes for multiple flexfields on the same METADATA line.

How You Load Data for Extensible Flexfields

This topic describes how to construct METADATA and MERGE lines in a data file to supply extensible flexfield data.
Category Code
Extensible flexfields have both a flexfield code and a category code. The category code is provided automatically in generated template files and has the attribute name EFF_CATEGORY_CODE.

Constructing the METADATA Line
Unlike descriptive flexfields, extensible flexfields aren't an extension of a business-object component. Rather, they're separate components in the business object hierarchy. To supply data for an extensible flexfield, you simply include the METADATA line for the extensible flexfield component in the data file. Remove the names of any attributes for which you aren't supplying data.

This example shows a METADATA line for the JobLegislative extensible flexfield.

```
METADATA|JobLegislative|EFF_CATEGORY_CODE|FLEX:PER_JOBS_LEG_EFF|JobId(SourceSystemId) |
JobCode|SetCode|EffectiveStartDate|EffectiveEndDate|SourceSystemOwner| 
SourceSystemId|LegislationCode|_EEO1_CATEGORY(PER_JOBS_LEG_EFF=HRX_US_JOBS) |
_FLSA_STATUS(PER_JOBS_LEG_EFF=HRX_US_JOBS) |_INSEE_PCS_CODE(PER_JOBS_LEG_EFF=FR) | 
_ECAP_JOB(PER_JOBS_LEG_EFF=FR) |_LINE_OF_PROGRESSION(PER_JOBS_LEG_EFF=HRX_US_JOBS)
```

Constructing the MERGE Lines
Each MERGE line for an extensible flexfield record can have only one context. In the METADATA line, you specify the context value against the flexfield code for the extensible flexfield.

This example shows a METADATA line and two MERGE lines for the JobLegislative extensible flexfield:

```
METADATA|JobLegislative|EFF_CATEGORY_CODE|FLEX:PER_JOBS_LEG_EFF|JobId(SourceSystemId) |
JobCode|SetCode|EffectiveStartDate|EffectiveEndDate|SourceSystemOwner| 
SourceSystemId|LegislationCode|_EEO1_CATEGORY(PER_JOBS_LEG_EFF=HRX_US_JOBS) |
_FLSA_STATUS(PER_JOBS_LEG_EFF=HRX_US_JOBS) |_INSEE_PCS_CODE(PER_JOBS_LEG_EFF=FR) | 
_ECAP_JOB(PER_JOBS_LEG_EFF=FR) |_LINE_OF_PROGRESSION(PER_JOBS_LEG_EFF=HRX_US_JOBS)
MERGE|JobLegislative|JOB_LEG|HRX_US_JOBS|OCT18EFF1|OCT18EFF1|COMMON|1990/01/01|4712/12
MERGE|JobLegislative|JOB_LEG|FR|OCT18EFF1|OCT18EFF1|COMMON|1990/01/01|4712/12/31|VISION|OCT18EFF1_LEG2|FR|| 
387b|N
```

Note: Unlike other components of the business-object hierarchy, extensible flexfields can't be supplied in isolation. They must be accompanied by a parent record.

Multirow Extensible Flexfield Contexts
You can configure an extensible flexfield context with multiple rows. In this case, you configure one or more of the flexfield segments as the key that uniquely identifies a single row of the multirow context. HCM Data Loader treats these flexfield segments as user-key attributes. Therefore, you can uniquely identify a single flexfield record using user keys.

You must supply a value for the user-key flexfield-segment attributes when creating an extensible flexfield record. When updating a multirow extensible-flexfield record, you must supply either the source key or the complete user key, including the flexfield segments defined as the unique key. You can identify which flexfield segments form the unique key by reviewing the flexfield-segment configuration on the Manage Extensible Flexfields page. All flexfield segments for which Unique Key is selected are user-key attributes.
Example of Loading Data for Extensible Flexfields

This topic shows how the METADATA line is constructed for a Job business object component with the configured JobLegislative extensible flexfield. The METADATA line appears automatically in the generated job template.

Constructing the METADATA Line for the JobLegislative Extensible Flexfield

The PER_JOBS_LEG_EFF extensible flexfield is configured in Oracle HCM Cloud with the context and attribute values shown in this table. The Hint column shows the resulting hint for each attribute.

<table>
<thead>
<tr>
<th>Context</th>
<th>Attribute</th>
<th>Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA</td>
<td>_EE0G</td>
<td>(PER_JOBS_LEG_EFF=CA)</td>
</tr>
<tr>
<td>CA</td>
<td>_NOC_CODE</td>
<td>(PER_JOBS_LEG_EFF=CA)</td>
</tr>
<tr>
<td>CH</td>
<td>_POSITION_TYPE</td>
<td>(PER_JOBS_LEG_EFF=CH)</td>
</tr>
<tr>
<td>FR</td>
<td>_ECAP_JOB</td>
<td>(PER_JOBS_LEG_EFF=FR)</td>
</tr>
<tr>
<td>FR</td>
<td>_INSEE_PCS_EXT_CODE</td>
<td>(PER_JOBS_LEG_EFF=FR)</td>
</tr>
</tbody>
</table>

In the generated Job.dat template file, the JobLegislative METADATA line includes these attribute definitions for the flexfield attributes:

```
METADATA|JobLegislative|...|EFF_CATEGORY_CODE|FLEX:PER_JOBS_LEG_EFF|_EE0G(PER_JOBS_LEG_DFF=CA)|
        |              |    |                |_NOC_CODE(PER_JOBS_LEG_DFF=CA)|_POSITION_TYPE(PER_JOBS_LEG_DFF=CH)|
        |              |    |                |_ECAP_JOB(PER_JOBS_LEG_DFF=FR)|_INSEE_PCS_EXT_CODE(PER_JOBS_LEG_DFF=FR)
```
8 Loading Setup Objects

Guidelines for Loading Absence Cases

You use absence cases to group related absences, typically those having the same cause. For example, a worker may take two periods of sickness absence after sustaining an injury at work. In this case, you can associate the two absences with the same absence case for ease of management. This topic describes how to load Absence Case objects for a worker using HCM Data Loader. Use the AbsenceCase.dat file.

Absence Categories

If you associate an absence case with an absence category, then all absences in the case must have the same category. If you specify no category for the absence case, then you can associate any type of absence with the absence case. You define absence categories using the Manage Absence Categories task in the Absence Administration work area.

Loading Absence Cases

These records must exist in Oracle HCM Cloud before you can create an absence case that refers to them:

- Absence category, if you’re using categories
- Worker and work relationship
- Worker assignment, if absences are specific to the assignment
- Absence records

Deleting Absence Cases

You can delete Absence Case objects using HCM Data Loader. To identify the record to be deleted, supply the AbsenceCaseCode attribute.

Examples of Loading Actions

Actions classify changes to data such as employment and compensation records. When you create or update a record, the action value specifies the reason for the change. For example, the associated action for an assignment change may be promotion or transfer. Action reasons, which are optional, provide additional information about a change. The Action Reason Usage component of an Actions object specifies the relationship between an action and an existing action reason. This topic shows how to load Actions objects and their Action Reason Usage components using HCM Data Loader.

Creating Actions

This example Actions.dat file creates an Actions component and associates it with an Action Reason Usage component. Both components are identified by source keys.
Loading Translated Action Names and Descriptions

Supply action names and descriptions in the language of the user who’s loading them. You supply an ActionsTranslation.dat file to translate action names and descriptions into other enabled languages once the actions exist. This example ActionsTranslation.dat file translates an action name and its description. The action is identified by its source key.

METADATA|ActionsTranslation|SourceSystemOwner|SourceSystemId|Language|ActionName|Description
MERGE|ActionsTranslation|VISION|ACT_VISION_PROM|FR|Performance|Changement est survenu en raison de la performance des employes

Deleting Actions

You can delete Actions and Action Reason Usage components using HCM Data Loader. This example Actions.dat file deletes an Action Reason Usage component. It must include a reference to the action to which it belongs. The Action Reason Usage component is identified by its source key.

Note: You can’t use user keys when deleting individual Action Reason Usage components.

When you delete an Actions component, you must also delete its Action Reason Usage component explicitly. This example Actions.dat file deletes an Actions component and its Action Reason Usage component. All DELETE instructions are in the same file. Both components are identified by source keys.

Examples of Loading Action Reasons

Action reasons, which are optional, provide additional information about an action. For example, the reason for promoting a worker may be good performance or seniority. This topic shows how to load action reasons using HCM Data Loader.

Creating Action Reasons

This example ActionReasons.dat file creates two Action Reasons components. The components are identified by source keys.
Loading Translated Action-Reason Names

Supply action-reason names in the language of the user who's loading them. You supply an ActionReasonsTranslation.dat file to translate action-reason names into other enabled languages once the action reasons exist. This example translates the names of existing action reasons. The Action Reasons components are identified by their source keys.

```
MERGE|ActionReasons|VISION|AR_VISION_PERF|VISION_PERF|Performance|2000/01/01|4712/12/31
MERGE|ActionReasons|VISION|AR_VISION_TXFR|VISION_TXFR|Internal Transfer|2000/01/01|
```

Deleting Action Reasons

You can delete Action Reasons components using HCM Data Loader. This example ActionReasons.dat file deletes two action reasons. It identifies the components using source keys.

```
METADATA|ActionReasons|SourceSystemOwner|SourceSystemId|ActionReason|Language
MERGE|ActionReasonsTranslation|VISION|AR_VISION_PERF|Accomplissement|FR
MERGE|ActionReasonsTranslation|VISION|AR_VISION_TXFR|Transfert interne|FR
```

Guidelines for Loading Calendar Events

A calendar event is an event, such as a public holiday or plant closure, on which working time may be affected. Calendar events are optional. You specify the geographical or organizational hierarchy to which workers affected by the calendar event must belong. The hierarchy must exist before you load the associated calendar events. This topic describes aspects of calendar events that you must understand to load them successfully using HCM Data Loader. It also provides examples showing how to load calendar events.

Calendar Event Lookup Categories

Calendar events have categories, which are defined in the PER_CAL_EVENT_CATEGORY lookup type. This lookup type has one delivered value, PH, for public holidays. If you use additional categories, then you must define them before you load calendar events. In the Setup and Maintenance work area, use the following:

- Functional Area: Workforce Information
- Task: Manage Availability Lookups

Calendar Event Coverage

The Calendar Event Coverage component identifies the branch of the geographical or organizational hierarchy to which the associated calendar event applies. You identify the top node of the hierarchy branch. The event applies to that node and its child nodes. You can also:

- Exclude from the coverage individual nodes that appear in the specified hierarchy branch.
- Override the calendar event name and its category for individual nodes in the hierarchy.
In a geographical hierarchy, the calendar event applies to all workers with assignments in the locations that you include in the calendar event coverage. In an organizational hierarchy, the calendar event applies to all workers with assignments in the departments that you include in the calendar event coverage. The affected workers may also have work schedules assigned to them. In this case, the event applies to the workers only if you add it as a resource exception to the work schedule or work schedule assignment.

Creating Calendar Events

This example CalendarEvent.dat file creates both a public holiday for Christmas Day and a half-day event for elapsed work schedules. It identifies the calendar events by their source keys.

```
METADATA|CalendarEvent|Name|Description|Category|CoverageType|ShortCode|StartDateTime|EndDateTime|TreeCode|TreeStructureCode|TreeVersionName|HalfDayForElapsed
MERGE|CalendarEvent|CPTAS6||PH|G|TAS6|2017/07/11 08:00:00|2017/07/11 12:30:00|WFMTL_Global|PER_GEO_TREE_STRUCTURE|WFMTL Bank Geography Version 1|Y
MERGE|CalendarEvent|Christmas Day 2017||PH|G|XMAS2017|2017/12/25 08:00:00|2017/12/25 18:00:00|WFMTL_Global|PER_GEO_TREE_STRUCTURE|WFMTL Bank Geography Version 1|N
METADATA|CalendarEventCoverage|ShortCode|CoverageScope|TerritoryCode|TreeStructureCode|TreeCode|TreeVersionName
MERGE|CalendarEventCoverage|TAS6|I|US|PER_GEO_TREE_STRUCTURE|WFMTL_Global|WFMTL Bank Geography Version 1
MERGE|CalendarEventCoverage|XMAS2017|I|US|PER_GEO_TREE_STRUCTURE|WFMTL_Global|WFMTL Bank Geography Version 1
```

Deleting Calendar Events

You can delete a calendar event unless it’s assigned to a work schedule. This example CalendarEvent.dat file deletes a calendar event. It identifies the calendar event by its source key.

```
METADATA|CalendarEvent|SourceSystemOwner|SourceSystemId
DELETE|CalendarEvent|VISION|XMAS2017
```

Guidelines for Loading Checklist Templates

You use a checklist template to define a sequence of related tasks with multiple performers, such as those for onboarding a new worker. For example, you can create a checklist template for new hires, with tasks for providing system access, issuing badges, allocating parking space, and so on. This topic describes how to create and maintain Checklist Template objects using HCM Data Loader.

Checklist Categories

The checklist category must exist in the CHECKLIST_CATEGORY lookup for the target environment. In the Setup and Maintenance work area, use the following to manage checklist categories:

- Functional Area: Workforce Information
- Task: Manage Checklist Lookups

The combination of checklist name and checklist category must be unique. **Enterprise Onboarding Step** checklists must either already exist in the target environment or be in the same .dat file as **Enterprise Onboarding** checklists that reference them.
Checklist Tasks

These rules apply to checklist tasks:

- Checklist task names must be unique for the checklist.
- If the current task has a preceding task, then the preceding task must be loaded before the current task can be created.
- When you create tasks of the type Configurable Form, the descriptive flexfield context must have been configured in the target environment.

Checklist Actions, Areas of Responsibility, and Eligibility Profiles

If you plan to use actions in the checklist template, then those actions must exist in the target environment.

If you plan to use areas of responsibility to identify task performers, then responsibility types must exist in the PER_RESPONSIBILITY_TYPES lookup in the target environment. Alternatively, you can use one of the values shown in this table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA_LN_MGR</td>
<td>Line manager</td>
</tr>
<tr>
<td>ORA_WORKER</td>
<td>Worker</td>
</tr>
<tr>
<td>ORA_INITIATOR</td>
<td>Initiator</td>
</tr>
</tbody>
</table>

The default performer is the line manager.

If you plan to use eligibility profiles in the checklist tasks, then those eligibility profiles must exist in the target environment.

Examples of Loading Checklist Templates

This topic provides examples showing how to create checklist templates using HCM Data Loader. These examples reference all components using user keys.

Creating Enterprise Onboarding Checklists

This example Checklist.dat file creates:

- A parent enterprise onboarding checklist template. It has:
  - A child checklist template
  - A single task
  - A Contents component
This example ChecklistContentDetails.dat file creates the content details for the Contents component of the parent checklist template.

COMMENT Create checklist content details
METADATA|ChecklistContentDetails|ContentCategory|ContentDefnCode|ContentSubtype|ContentType|Title|Status
MERGE|ChecklistContentDetails|ORA_ONB_INSIGHT|CHKLIST_CONT_DEF67|ORA_ONB_EVENT_STANDARD|ORA_ONB_EVENT|Event65|A

Creating Standard Onboarding Checklists

This example Checklist.dat file creates a standard onboarding checklist template. It has:

- Two tasks, which must be performed in the specified order
- A notification override for one of the tasks
- A Contact component

COMMENT Create the checklist template
METADATA|Checklist|ChecklistName|Country|ChecklistCategory|AllocatedOn|CompletedOn|OffsetDays
MERGE|Checklist|Checklist_Onboard_1|United States|ONBOARD|ORA_CHK_ALLOCATION_DT|ORA_CHK_MAND_TASK_COMPLETE|0
Related Topics

• Enterprise Onboarding Checklists

• Considerations for Creating Checklist Templates

Guidelines for Loading Document Delivery Preferences

Employers periodically deliver documents, such as payslips and year-end tax statements, to workers. Document delivery preferences specify how workers receive those documents. For example, workers may receive their payslips online. You can specify a default delivery method for a document type, and you can override the default method on relevant work structures. For example, delivery preferences for payroll documents can be overridden at Payroll Statutory Unit level. You can also specify delivery preferences for a person. Preferences specified at person level override those at all other levels. This topic describes how to load Document Record Delivery Preference objects for a person using HCM Data Loader.

Document Type

Before you can load document delivery preferences for a person:

• The document type must exist in the target environment.

• Document delivery preferences must be enabled for the document type. In the document type definition, **Override Hierarchy** must be set to either **Payroll** or **General**, as appropriate. For example, if you’re loading delivery preferences for performance documents, then **Override Hierarchy** must be set to **General**.

Loading Document Delivery Preferences


This example DocumentDeliveryPreference.dat file loads document delivery preference records. These records are for the document type identified by source key DT1345 and the three people identified by the person number user key.

```
METADATA|DocumentDeliveryPreference|DocumentTypeId(SourceSystemId)|LevelCode|PersonNumber|OnlineEnabledFlag|PaperEnabledFlag|AllowWorkerOverrideFlag|OnlineConsentRequiredFlag|InitialConsentFlag|SourceSystemId|SourceSystemOwner
MERGE|DocumentDeliveryPreference|DT1345|900_PERSON|Z8154257|Y|N|Y|Y|Y|LoadPref1|PSFT-US
MERGE|DocumentDeliveryPreference|DT1345|900_PERSON|Z8154806|N|Y|Y|Y|Y|LoadPref3|PSFT-US
```
Guidelines for Loading Document Types

A document type, such as Performance Improvement Plan or Letter of Recognition, defines the purpose and treatment of documents of that type. When defining a document type, you include attributes, such as the document name, dates, and issuing authority, and specify whether they’re required. You can also specify an expiration notification period, indicate whether approvals are required, and enable multiple occurrences of the document. This topic describes aspects of the Document Type object that you must understand to load document types successfully using HCM Data Loader.

Document Categories and Subcategories

Document types belong to a document category and may also belong to a document subcategory. Document categories, such as Expenses or Payroll, provide a way to group document types for ease of retrieval and management. Document subcategories, such as Involuntary Deduction or Additional Income in the Payroll category, provide a further level of detail.

Before loading document types, ensure that referenced document categories and subcategories exist in the target environment. In the Setup and Maintenance work area, use the following to create document categories:

- Functional Area: Workforce Information
- Task: Manage Document Lookups

In the Setup and Maintenance work area, use the following to create document subcategories:

- Functional Area: Workforce Information
- Task: Manage Extended Lookup Codes

Document Types

When loading document types, you must provide a unique reference for the record being created. If your document type isn’t country-specific, then you must supply a source key. This rule exists because the Country attribute forms part of the user key.

If you’re loading a Document Type Delivery Preference component, then set the Hierarchy Code attribute of the Document Type component to either GENERAL or PAYROLL. If you omit this attribute or provide no value, then document delivery preferences aren’t enabled for the document type.

Document Type Delivery Preferences

The Document Type Delivery Preference component enables users to specify delivery preferences for documents of the associated type. For example, you could enable users to specify that payslips can be delivered both on paper and online.

The delivery preferences for the document type can be overridden. That is, you can specify document delivery preferences:

- At the payroll statutory unit level, for documents in the PAYROLL category
- At the legal employer level, for documents in all other categories

In both cases, delivery preferences can be further overridden at department and location levels.
This figure summarizes the override hierarchy for document delivery preferences. Entries at any level of the hierarchy override those above them.

If you set the **Allow Worker Override** attribute to **Y**, then workers can specify document delivery preferences for themselves. Worker preferences override preferences at all other levels.

**Related Topics**
- How You Set Preferences for Document Delivery
- Document Types and Categories
Examples of Loading Document Types

This topic provides examples showing how to load and manage Document Type objects using HCM Data Loader.

Creating Document Types Using Source Keys

This example DocumentType.dat file creates one global and one country-specific document type. The document types are identified using source keys.

METADATA|DocumentType|SourceSystemOwner|SourceSystemId|DocumentType|Description|LegislationCode|CategoryCode|AuthorizationRequiredFlag|MultipleOccurencesFlag|ActiveInactiveFlag|PublishRequiredFlag|HierarchyCode|DocumentNameRequired|DocumentNumberRequired|DateFromRequired|DateToRequired|IssuingCountryRequired|IssuingLocationRequired|IssuingAuthorityRequired|IssuingDateRequired|CommentsRequired

MERGE|DocumentType|VISION|DTYPETest001|RS Global DocType1|RS Global DocType1 Description||PAYROLL|Y|Y|N|Y|GENERAL|R|R|Y|Y|Y|Y|Y|Y|Y|Y
MERGE|DocumentType|VISION|DTYPETest003|RS US DocType1|RS US DocType1 Description|US|AUDIT|Y|Y|N|Y|PAYROLL|R|R|Y|Y|Y|Y|Y|Y|Y|Y

Creating Document Types Using User Keys

This example DocumentType.dat file creates one country-specific document type. The document type is identified using user keys.

METADATA|DocumentType|Description|Country|CategoryCode|AuthorizationRequiredFlag|MultipleOccurencesFlag|ActiveInactiveFlag|PublishRequiredFlag|HierarchyCode|DocumentNameRequired|DocumentNumberRequired|DateFromRequired|DateToRequired|IssuingCountryRequired|IssuingLocationRequired|IssuingAuthorityRequired|IssuedDateRequired|CommentsRequired

MERGE|DocumentType|US Audit Doc Type|US Audit Doc Type Description|United States|AUDIT|Y|Y|Y|Y|PAYROLL|R|R|Y|Y|Y|Y|Y|Y

Note: If the document type were global, you would have to provide a source key rather than a user key. This requirement exists because Country is a required attribute of the user key.

Creating Document Types with Delivery Preferences

This example DocumentType.dat file creates a country-specific document type with delivery preferences for the document type, payroll statutory unit, and department. The document type and delivery preferences are identified using source keys.

METADATA|DocumentType|SourceSystemOwner|SourceSystemId|DocumentType|Description|LegislationCode|CategoryCode|AuthorizationRequiredFlag|MultipleOccurencesFlag|ActiveInactiveFlag|PublishRequiredFlag|HierarchyCode|DocumentNameRequired|DocumentNumberRequired|DateFromRequired|DateToRequired|IssuingCountryRequired|IssuingLocationRequired|IssuingAuthorityRequired|IssuingDateRequired|CommentsRequired

MERGE|DocumentType|VISION|DTYPETest003|RS US DocType1|RS US DocType1 Description|US|AUDIT|Y|Y|N|Y|PAYROLL|R|R|Y|Y|Y|Y|Y|Y|Y|Y
MERGE|DeliveryPreference|SourceSystemOwner|SourceSystemId|DocumentTypeId(SourceSystemId)|InitialConsentValueFlag|OnlineConsentRequiredFlag|OnlineEnabledFlag|PaperEnabledFlag|AllowWorkerOverrideFlag|EmailEnabledFlag|LegislationCode|LevelCodeName|DocumentTypeCountry|DocumentTypeLegislationCode|PayrollStatutoryUnitName|DepartmentName
Loading Translated Document Types and Descriptions

This example DocumentTypeTranslation.dat file translates an existing document type and its description. It identifies the document type using source keys.

```
MERGE|DeliveryPreference|VISION|DTYPETest003_Pref0|DTYPETest003|Y|N|Y|N|US|US||US
MERGE|DeliveryPreference|VISION|DTYPETest003_Pref1|DTYPETest003|Y|N|Y|N|US|Payroll Statutory Unit|US|US|
MERGE|DeliveryPreference|VISION|DTYPETest003_Pref2|DTYPETest003|Y|N|Y|N|US|Department|US|US|GBI HCM Widgets USA|HCM-1001-Corporate
```

Deleting Document Types

You can delete a document type only if no document records of that type exist. When you delete a document type, any associated delivery preferences are also deleted. This example DocumentType.dat file deletes a document type that is identified by its source key.

```
DELETE|DocumentType|VISION|DTYPETest001
DELETE|DocumentType|VISION|DTYPETest003
```

This example DocumentType.dat file deletes a document type that is identified by its user key.

```
DELETE|DocumentType|US Audit Doc Type|US Audit Doc Type Description|United States
```

Examples of Loading Extended Lookup Codes

You use extended lookup codes to provide subcategories for lookup codes. The associated lookup codes must already exist when you load Extended Lookup objects using HCM Data Loader. If the lookup codes are for specific legislations, then you can supply extended lookup codes for those legislations only. For example, if a lookup code has the +FR tag, then you can supply extended lookup codes for the FR legislation code only. This topic provides examples showing how to load Extended Lookup objects using HCM Data Loader.

Creating Extended Lookup Codes

This example ExtendedLookupCode.dat file creates extended lookup codes for the CONTRACT_TYPE lookup type. It identifies the extended lookup codes using source keys.

```
MERGE|ExtendedLookupCode|VISION|ELC_CONTRACT_LIMITED|CONTRACT_TYPE|5|NO|L|Limited Contract
```
Loading Translated Extended Lookup Code Names

Supply the names of extended lookup codes in the language of the user who’s loading them. You supply an ExtendedLookupCodeTranslation.dat file to translate the names of extended lookup codes into other enabled languages once the codes exist. This example translates the name of an existing extended lookup code. It identifies the code by its source key.

METADATA|ExtendedLookupCodeTranslation|SourceSystemOwner|SourceSystemId|Language|ExtendedLookupCodeName
MERGE|ExtendedLookupCodeTranslation|VISION|ELC_CONTRACT_LIMITED|FR|Contrat a Duree Limitee

Deleting Extended Lookup Codes

You can delete Extended Lookup objects using HCM Data Loader. This example ExtendedLookupCode.dat file deletes specific extended lookup codes for the CONTRACT_TYPE lookup code. It identifies the extended lookup codes using source keys.

METADATA|ExtendedLookupCode|SourceSystemOwner|SourceSystemId
DELETE|ExtendedLookupCode|VISION|ELC_CONTRACT_LIMITED
DELETE|ExtendedLookupCode|VISION|ELC_CONTRACT_DIRECTOR

Guidelines for Loading Name Formats

A name format is a set of rules for combining individual name components, such as first name and last name, to form a complete person name. Name formats are specific to a legislation and name-format type. The name-format types are display name, list name, order name, and full name. The predefined global format is used when no format exists for a format type and legislation. This topic describes how to load Name Format objects using HCM Data Loader.

Understanding Format Masks

A format mask is a string of codes used to construct a name format. The codes represent the required name components, symbols, and special characters. Use the codes shown in this table to identify the name components.

<table>
<thead>
<tr>
<th>Code</th>
<th>Name Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>$FIR$</td>
<td>First Name</td>
</tr>
<tr>
<td>$LAS$</td>
<td>Last Name</td>
</tr>
<tr>
<td>$MID$</td>
<td>Middle Name</td>
</tr>
<tr>
<td>$PLN$</td>
<td>Previous Last Name</td>
</tr>
<tr>
<td>$KNA$</td>
<td>Known As</td>
</tr>
<tr>
<td>Code</td>
<td>Name Component</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>$HNNS$</td>
<td>Honors</td>
</tr>
<tr>
<td>$PNAS$</td>
<td>Prefix</td>
</tr>
<tr>
<td>$SUF$</td>
<td>Suffix</td>
</tr>
<tr>
<td>$TIT$</td>
<td>Title</td>
</tr>
<tr>
<td>$MLR$</td>
<td>Military Rank</td>
</tr>
<tr>
<td>$INF1$  through $INF30$</td>
<td>Name Information 1 through Name Information 30</td>
</tr>
</tbody>
</table>

Use the codes shown in this table for symbols and special characters.

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$SPA$</td>
<td>space</td>
<td>space character ( )</td>
</tr>
<tr>
<td>$COM$</td>
<td>comma</td>
<td>,</td>
</tr>
<tr>
<td>$OPE$</td>
<td>left parenthesis</td>
<td>(</td>
</tr>
<tr>
<td>$CLO$</td>
<td>right parenthesis</td>
<td>)</td>
</tr>
<tr>
<td>$QUO$</td>
<td>quotation mark</td>
<td>*</td>
</tr>
<tr>
<td>$DOT$</td>
<td>period</td>
<td>.</td>
</tr>
<tr>
<td>$SLA$</td>
<td>slash</td>
<td>/</td>
</tr>
<tr>
<td>$COL$</td>
<td>colon</td>
<td>:</td>
</tr>
<tr>
<td>$SEM$</td>
<td>semicolon</td>
<td>;</td>
</tr>
<tr>
<td>$ATT$</td>
<td>at sign</td>
<td>@</td>
</tr>
</tbody>
</table>

**Constructing a Format Mask**

The format mask must:

- Start and end with a vertical bar (|).
- Separate each name component with two vertical bars (||).
The vertical bar is the default delimiter for HCM Data Loader .dat files. If you haven’t selected a different default delimiter, then you must prefix the vertical bar in the name-format mask with the HCM Data Loader escape character. The escape character ensures that HCM Data Loader ignores delimiters in the format mask. The default escape character is the backslash (\). For example, to provide a format mask for the name format Title Last Name, First Name (Known As), you supply the codes for each element of the name as shown in this table.

<table>
<thead>
<tr>
<th>Element</th>
<th>Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title space</td>
<td>$TIT$$SPA$</td>
</tr>
<tr>
<td>Last Name, space</td>
<td>$LAS$$COM$$SPA$</td>
</tr>
<tr>
<td>First Name space</td>
<td>$FIR$$SPA$</td>
</tr>
<tr>
<td>(Known As)</td>
<td>$OPE$$KNA$$CLO$</td>
</tr>
</tbody>
</table>

In the format mask, each name component must be delimited by two vertical bars:

$TIT$$SPA$\|$LAS$$COM$$SPA$\|$FIR$$SPA$\|$OPE$$KNA$$CLO$

In addition, the format mask must start and end with a single vertical bar:

\|$TIT$$SPA$\|$LAS$$COM$$SPA$\|$FIR$$SPA$\|$OPE$$KNA$$CLO$\|

Creating Name Formats

This example NameFormat.dat file creates a display name for France in the format Title First Name Last Name. The name format is identified by its source key.

METADATA|NameFormat|SourceSystemOwner|SourceSystemId|FormatName|LegislationCode|UserFormatChoice|FormatMask
MERGE|NameFormat|VISION|NF_FR_L_DISP|DISPLAY_NAME|FR|L|\|$TIT$$SPA\|$\|$FIR$$SPA\|$\|$LAS$

Deleting Name Formats

You can delete a Name Format object using HCM Data Loader, provided that the name format isn’t being used. This example NameFormat.dat file deletes a name format. It identifies the name format by its source key.

METADATA|NameFormat|SourceSystemOwner|SourceSystemId
DELETE|NameFormat|VISION|NF_FR_L_DISP

Examples of Loading Person Types

System person types are predefined values that identify groups such as employees and contingent workers. You can’t create, edit, or delete system person types. However, each system person type is associated with one or more user person types, which further categorize the group. You can create, edit, and delete user person types. For example, you could define Associate and Remote Worker as user person types of the Employee system person type. For any system person type, one user person type must be identified as the default value. This topic provides examples showing how to load and manage user Person Type objects using HCM Data Loader.
Creating Person Types

This example PersonType.dat file creates the Officer and Rating user person types for the Employee system person type. It identifies the person types using source keys.

```
METADATA|PersonType|SourceSystemOwner|SourceSystemId|SystemPersonType|UserPersonType|ActiveFlag|DefaultFlag
MERGE|PersonType|VISION|PT_EMP_OFFICER|EMP|Officer|Y|N
MERGE|PersonType|VISION|PT_EMP_RATING|EMP|Rating|Y|N
```

Loading Translated Person-Type Names

Supply person-type names in the language of the user who’s loading them. You supply a PersonTypeTranslation.dat file to translate person-type names into other enabled languages once the person types exist. This example translates the name of an existing person type. The person type is identified by its source key.

```
METADATA|PersonTypeTranslation|SourceSystemOwner|SourceSystemId|Language|UserPersonType
MERGE|PersonTypeTranslation|VISION|PT_EMP_OFFICER|FR|Officier
```

Deleting Person Types

You can delete Person Type objects using HCM Data Loader, provided that the person type isn’t being used. This example PersonType.dat file deletes person types and any translated versions of the person-type names. It identifies the person types by their source keys.

```
METADATA|PersonType|SourceSystemOwner|SourceSystemId
DELETE|PersonType|VISION|PT_EMP_OFFICER
DELETE|PersonType|VISION|PT_EMP_RATING
```

Examples of Loading Resource Exceptions

A resource exception is a deviation in availability from a work schedule or schedule assignment. The resource exception defines when a resource is unavailable. For example, a worker may be attending training and therefore unavailable between specified dates. You create resource exceptions, which are optional, for specific work schedules or schedule assignments. The work schedule or schedule assignment must exist before you can create an associated resource exception. This topic provides examples showing how to load Resource Exception objects using HCM Data Loader.

Creating Resource Exceptions

This example ResourceException.dat file creates a resource exception for a hospital appointment. It identifies the resource exception by its source key.

```
METADATA|ResourceException|SourceSystemOwner|SourceSystemId|ExceptionName|StartDateTime|EndDateTime
MERGE|ResourceException|VISION|RE_VISION_HOSPITAL|Hospital Appointment|2015/08/15 08:00:00|2015/08/15 17:00:00
```
Deleting Resource Exceptions

You can delete a Resource Exception object using HCM Data Loader, provided that it’s not referred to by a work schedule. This example ResourceException.dat file deletes an unused resource exception. It identifies the resource exception by its source key.

METADATA|ResourceException|SourceSystemOwner|SourceSystemId
DELETE|ResourceException|VISION|RE_VISION_HOSPITAL

Guidelines for Loading Time and Labor Event Groups

An event group is a group of related events, such as changes to a worker’s assignment, that require a retroactive recalculation of time cards. This topic explains some aspects of event groups that you must understand to load them successfully using HCM Data Loader.

Loading Event Groups

These rules apply to the Event Group component:

- You can set EventGroupCode and EventGroupName to any user-defined value that identifies the purpose of the group.
- To resubmit time cards, you must set EventGroupType to A (Action).

These rules apply to the Date Tracked Event, Event Value Change, and Event Value Qualifier components:

- If you set UpdateType to DT_INSERT, then leave ColumnName blank.
- EventGroupCode identifies the group to which the component belongs.

These rules apply to the Event Value Change Component:

- Sequence specifies the sequence in which qualifying conditions are loaded. The sequence is important, as conditions can exist in a hierarchical relationship to each other. The sequence number also provides a way for other nodes in the condition hierarchy to refer to a condition.
- ValidEvent must be Y or N. It specifies whether the condition qualifies or disqualifies the event. For example, your qualifying condition may specify that an event is valid for any location change, except when moving from New York to San Francisco. In this case, you define two rows as shown in this table:

<table>
<thead>
<tr>
<th>Condition</th>
<th>From Value</th>
<th>To Value</th>
<th>Valid Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent</td>
<td>Any</td>
<td>Any</td>
<td>Y</td>
</tr>
<tr>
<td>Child</td>
<td>New York</td>
<td>San Francisco</td>
<td>N</td>
</tr>
</tbody>
</table>

- FromValue and ToValue can be any valid value for the column or <ANY VALUE> to indicate that any value can trigger resubmission of time cards.
- ParentEvtValChangeSequence is the sequence number of the parent Event Value Change component in the condition hierarchy.
These rules apply to the Event Value Qualifier component:

- **Sequence** is the sequence of the topmost Event Value Change component in the value change hierarchy for given Date Tracked Event and Event Value Qualifier components.
- **QualifierName** is the qualifier name, which is predefined in the PAY_EVENT_QUALIFIERS_F table.
- **QualifierValue** must be Y or N. If you set this value to Y, then updates to the worker’s primary assignment trigger the event. If you set this value to N, then updates to the worker’s secondary assignments trigger this event.

**Examples of Loading Time and Labor Event Groups**

An event group is a group of related events, such as changes to a worker’s assignment, that require a retroactive recalculation of time cards. This topic shows how to create event groups using HCM Data Loader.

**Loading Event Groups**

This example EventGroup.dat file creates a single Event Group component, which is identified by its user key. For resubmitting time cards, the **EventGroupType** attribute value must be A.

```
METADATA|EventGroup|EventGroupCode|EventGroupName|EventGroupType
MERGE|EventGroup|JobChangeEventGroup|JobChangeEventGroup|A
```

**Loading Date Tracked Events**

The Date Tracked Event component identifies the type of event (update, correction, or insert) that triggers resubmission of time cards. It also identifies the affected object and the affected column or value in that object. This example identifies the Date Tracked Event component by its user key.

```
METADATA|DateTrackedEvent|ColumnName|UpdateType|EventGroupCode|DatedObjectName
MERGE|DateTrackedEvent|JOB_ID|DT_UPDATE_COLUMN|JobChangeEventGroup|
oracle.apps.hcm.employment.core.publicModel.entity.EmployeeAssignmentDEO
```

**Loading Event Value Changes**

The Event Value Change component specifies the value changes that can cause time cards to be resubmitted. For example, the Date Tracked Event component may be monitoring changes to a worker’s job. In that case, the Event Value Change component could specify the current and new jobs that trigger resubmission of time cards. This example identifies the Event Value Change component by its user key.

```
METADATA|EventValueChange|ColumnName|UpdateType|EventGroupCode|DatedObjectName|Sequence|ValidEvent|FromValue|ToValue|EffectiveStartDate|EffectiveEndDate
MERGE|EventValueChange|JOB_ID|DT_UPDATE_COLUMN|JobChangeEventGroup|
oracle.apps.hcm.employment.core.publicModel.entity.EmployeeAssignmentDEO|1|Y|<ANY_VALUE>|<ANY_VALUE>|1950/01/01|4712/12/31
```
Loading Event Value Qualifiers

Using the optional Event Value Qualifier component, you can qualify an event. For example, when monitoring job changes, you could use this component to specify that only job changes in primary assignments are of interest. This example identifies the Event Value Qualifier component by its user key.

```
METADATA|EventValueQualifier|ColumnName|UpdateType|EventGroupCode|DatedObjectName|Sequence|EffectiveStartDate|EffectiveEndDate|QualifierName|QualifierValue
MERGE|EventValueQualifier|JOB_ID|DT_UPDATE_COLUMN|JobChangeEventGroup|oracle.apps.hcm.employment.core.publicModel.entity.EmployeeAssignmentDEO|1|1950/01/01|4712/12/31|EmployeeAssignmentDEO Primary Flag|Y
```

Columns and Event Source Objects for Time and Labor Events

When defining a Time and Labor event, you must provide values for the Column Name (ColumnName) and Event Source Object Name (DatedObjectName) attributes. This topic lists the values of these attributes for each Time and Labor event.

<table>
<thead>
<tr>
<th>Event</th>
<th>Column Name</th>
<th>Event Source Object Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marital Status</td>
<td>MARITAL_STATUS</td>
<td>oracle. apps. hcm. people. core. protectedModel. entity. PersonLegislativeInfoDEO</td>
</tr>
<tr>
<td>Home Location</td>
<td>ADDRESS_ID</td>
<td>oracle. apps. hcm. addresses. publicModel. entity. AddressDEO</td>
</tr>
<tr>
<td>Effective Start Date</td>
<td>EFFECTIVE_ START_DATE</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Effective End Date</td>
<td>EFFECTIVE_ END_DATE</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Business Unit</td>
<td>BUSINESS_ UNIT_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Grade Ladder</td>
<td>GRADE_ LADDER_PGM_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Working Hours</td>
<td>NORMAL_HOURS</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Event</td>
<td>Column Name</td>
<td>Event Source Object Name</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Standard Working Hours</td>
<td>TOTAL_HOURS</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. WorkingHourPatternDEO</td>
</tr>
<tr>
<td>Job</td>
<td>JOB_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Grade</td>
<td>GRADE_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Is Manager</td>
<td>MANAGER_FLAG</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Location</td>
<td>LOCATION_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Position</td>
<td>POSITION_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Worker Category</td>
<td>EMPLOYEE_CATEGORY</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Assignment Category</td>
<td>EMPLOYMENT_CATEGORY</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Regular or Temporary</td>
<td>PERMANENT_TEMPORARY_FLAG</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Full-Time or Part-Time</td>
<td>FULL_PART_TIME</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Person Type</td>
<td>PERSON_TYPE_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Hourly Paid or Salaried</td>
<td>HOURLY_SALARIED_CODE</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Frequency</td>
<td>FREQUENCY</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Event</td>
<td>Column Name</td>
<td>Event Source Object Name</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Working at Home</td>
<td>WORK_AT_HOME</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Legal Employer</td>
<td>LEGAL_ENTITY_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Department</td>
<td>ORGANIZATION_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Reporting Establishment</td>
<td>ESTABLISHMENT_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Bargaining Unit</td>
<td>BARGAINING_UNIT_CODE</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Collective Agreement</td>
<td>COLLECTIVE_AGREEMENT_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Union Member</td>
<td>LABOR_UNION_MEMBER_FLAG</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Overtime Period</td>
<td>OVERTIME_PERIOD</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
<tr>
<td>Manager</td>
<td>MANAGER_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. AssignmentSupervisorDEO</td>
</tr>
<tr>
<td>Work Day Definition</td>
<td>WORK_DAY_DEF_ID</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. WorkingHourPatternDEO</td>
</tr>
<tr>
<td>Assignment DFFs</td>
<td>Provide the column name for the global context DFF.</td>
<td>oracle. apps. hcm. employment. core. publicModel. entity. EmployeeAssignmentDEO</td>
</tr>
</tbody>
</table>

### Examples of Loading Time and Labor Event Actions

An event action defines how to react to an event, for example, whether to recalculate time cards when a worker’s job or location changes. This topic shows how to create event actions using HCM Data Loader.
Loading Event Actions

This example EventAction.dat file creates a single Event Action component, which is identified by its user key.

```
METADATA|EventAction|EventActionCode|EffectiveStartDate|EffectiveEndDate|EventActionName|EventActionTypeCode|
LookbackTimeDefinitionCode|ActionSubmission|ProcessMode|AutoApprove
MERGE|EventAction|JobChangeAction|1950/01/01|4712/12/31|JobChangeAction|ORA_HWM_ACTION_TYPE_TC_RESUBMIT|
12_MONTHS_AGO_MONTH_START_DATE|SYNC|MANUAL|Y
```

These rules apply:

- To resubmit time cards, you must set `EventActionTypeCode` to `ORA_HWM_ACTION_TYPE_TC_RESUBMIT`.
- The `LookbackTimeDefinitionCode` value identifies a time period before the action execution date. Time cards from this period are considered for resubmission. Use one of these values:
  - `3_MONTHS_AGO_MONTH_START_DATE`
  - `6_MONTHS_AGO_MONTH_START_DATE`
  - `12_MONTHS_AGO_MONTH_START_DATE`

For example, a job change is effective on 1st January 2018 and the time-card resubmission process runs on that date. If `LookbackTimeDefinitionCode` is `3_MONTHS_AGO_MONTH_START_DATE`, then time cards from 1st October 2017 are considered for resubmission.

- `ActionSubmission` must be `SYNC` for time-card resubmission.
- `ProcessMode` determines whether time cards are resubmitted automatically by the process or marked for resubmission but manually resubmitted. Set it to `AUTO` or `MANUAL`, as appropriate.
- `AutoApprove` determines whether time cards are approved automatically after they're resubmitted. Valid values are `Y` and `N`.

Loading Event Action Criteria

The Event Action Criteria component associates an event action with a specific group of workers. This Event Action Criteria component is identified by its user key.

```
METADATA|EventActionCriteria|EventActionCode|EffectiveStartDate|EffectiveEndDate|HCMGroupCode
MERGE|EventActionCriteria|JobChangeAction|1950/01/01|4712/12/31|ABC_GRP
```

`HCMGroupCode` values are defined in the GRP_CODE column of the HWM_GRP_VL table view and can be accessed using SQL. You can omit the Event Action Criteria component if the action applies to all workers.

Loading Event Group Actions

The Event Group Action component associates an event action with an event group. This Event Group Action component is identified by its user key.

```
METADATA|EventGroupAction|EventActionCode|EffectiveStartDate|EffectiveEndDate|EventGroupCode
MERGE|EventGroupAction|JobChangeEventGroup|1950/01/01|4712/12/31|JobChangeEventGroup
```
Related Topics

- Tables and Views for Oracle HCM Cloud
9 Loading Workers

Overview of Loading Workers

The Worker object includes all information related to a person and his or her work relationships. This topic describes some general considerations that apply when you load workers.

The Worker Hierarchy

Select the Worker object on the View Business Objects page to review its component hierarchy:

Using Source Keys

You can’t update most components of the worker hierarchy if you supply only a user key. This restriction exists because the attribute that you want to change is the attribute that’s used to identify the record. For example, in the Person Address component, the AddressLine1 attribute is used both to identify the address to update and to supply the new value. Therefore, you’re recommended always to supply source keys when creating workers and use them when updating worker records.

Multiple Instances of a Component

A person can have multiple instances of some components, such as Person Address, Person Phone, and Person Email. When you load multiple instances of a component for a person, you must:

- Identify one of the records as primary using the PrimaryFlag attribute of the component.
- Process the occurrences together and include the parent Worker component in the file. If the worker already exists in Oracle HCM Cloud, then you can include just the primary-key attributes of the worker.

The Employment Model

You must understand the employment model in the legal employer to which you’re loading work relationships and assignments. In any legal employer, the Employment Model option can be set to one of these values:

- 2 Tier - Single Assignment
- 2 Tier - Multiple Assignment
- 2 Tier - Single Contract - Single Assignment
- 2 Tier - Multiple Contract - Single Assignment
- 3 Tier - Single Employment Terms - Single Assignment
- 3 Tier - Single Employment Terms - Multiple Assignment
• 3 Tier - Multiple Employment Terms - Single Assignment
• 3 Tier - Multiple Employment Terms - Multiple Assignment

Regardless of the employment model, an employment terms record is always created for an assignment. Therefore, you must include an Employment Terms component in the .dat file. When loading multiple work relationships or assignments for a person, you must identify which is primary using the PrimaryFlag attribute. A person must have only one primary work relationship at a time and only one primary assignment in each work relationship.

Defining Referenced Values

Many components of the Worker object include values, such as person type and assignment status, that must exist in the target environment. Perform the tasks shown in this table to define relevant values before you load data.

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage Actions</td>
<td>Defines the actions used to classify changes to employment data</td>
</tr>
<tr>
<td>Manage Person Types</td>
<td>Defines subcategories of the predefined person types, such as Employee and Nonworker</td>
</tr>
<tr>
<td>Manage Assignment Status</td>
<td>Defines status values, such as active, inactive, or suspended, for assignments</td>
</tr>
</tbody>
</table>

In addition, you must have reviewed and updated lists of values, such as address types, phone types, ethnicity, and marital status, before loading workers. You may have performed this step during implementation. If you’re synchronizing assignments from positions, then you must enable position synchronization before you load assignments.

Worker Termination

You terminate work relationships, not workers. When you terminate a work relationship using HCM Data Loader, its child components, such as Assignment components, are terminated automatically. Don’t try to terminate other child components of the worker object, such as Person Name. The person record must continue to exist and be returned in search results, for example.

Loading the Person Attributes Descriptive Flexfield

The Person Attributes (PER_PERSONS_DFF) descriptive flexfield isn't date-effective, and a person can have only one context value at a time. If you load a new context value, then it overwrites any existing context value.

Related Topics

• Employment Model
• Source Keys
• How You Review Lists of Values
• Sources of Business-Object Information
Guidelines for Loading New Workers

The Worker object hierarchy is complex, with many components and hierarchy levels. When you create workers, especially in large volumes, you’re recommended to load only the required components initially. Once the Worker objects exist, you can load related objects, such as Salary and Element Entry. This topic identifies the components that you must load when you create workers and those that you must load in some circumstances. You can load optional Worker components once the Worker objects exist.

Loading Required Components of the Worker Object

Supply these components of the Worker object when you create workers using HCM Data Loader:

- Worker
- Person Legislative Data
- Person Name
- Work Relationship
- Employment Terms
- Assignment

Include date-effective history, as appropriate, for each worker.

Note: The worker StartDate and EffectiveStartDate attribute values must both be equal to the start date of the worker’s earliest work relationship.

Loading Person Legislative Data

Person Legislative Data isn’t a required component for creating workers. However, you’re recommended to include it and provide a source-key value. Otherwise, the component is created automatically with a default source-key value.

Loading Assignment Managers

Load Assignment Manager components to create manager relationships only after you have loaded workers successfully. This approach ensures that the worker records of referenced managers exist before you refer to them in Assignment Manager components.

Loading Contracts

You can supply contract details when using an employment model that supports contracts. You must supply the Contract component when creating workers, if appropriate. You can’t load Contract components later.

Loading Users and Roles

The enterprise option User Account Creation controls whether user accounts are created automatically for workers loaded in bulk. However, you can prevent user accounts being created for individual workers by including the

• Position Synchronization
**GeneratedUserAccountFlag** attribute of the User Information component and setting it to N. You can also specify a user name on the **UserName** attribute of the User Information component. Otherwise, the user name is generated automatically in the enterprise default format. If you set one of the following attributes or attribute pairs, then an attempt is made to link an existing user account to the worker:

- **UserGUID**
- **UserName** and **UsernameMatchingFlag**
- **EmailAddress** and **EmailMatchingFlag**

Roles are provisioned to new users automatically, as specified by current role-provisioning rules. To assign roles manually to a new worker, include the Person User Manual Roles component with the Worker object.

**Note:** You can supply the User Information and Person User Manual Roles components only for new workers and only at the time you create them. Once a worker exists, you use the User object and its User Role component to manage users and roles.

If user accounts are created by default in your environment, then you can prevent user accounts being created for terminated workers. Set the enterprise option **User Account Creation for Terminated Workers** to No. When you load terminated workers in bulk, this setting prevents the generated user-account requests from being processed. In the Setup and Maintenance work area, use the following to set this option:

- Functional Area: Workforce Structures
- Task: Manage Enterprise HCM Information

### Effective Start and End Dates for Worker Components

Many components of the Worker object are date effective. When creating workers, you must ensure that the effective dates of individual worker components don’t conflict. This topic suggests an approach to setting the earliest effective start date and last effective end date for each date-effective component. You don’t have to follow this guidance. However, you must ensure that effective dates are aligned for the complete Worker object.

This table suggests how to set earliest effective start dates and last effective end dates for date-effective worker components.

<table>
<thead>
<tr>
<th>Component</th>
<th>Earliest Effective Start Date</th>
<th>Last Effective End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worker</td>
<td>Earliest start date of the worker.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Person Address</td>
<td>On or after the earliest start date of the worker.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Person Legislative Data</td>
<td>Earliest start date of the worker.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Person Name</td>
<td>Earliest start date of the worker.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Person Visa</td>
<td>On or after the earliest start date of the worker.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Person Contact Relationship</td>
<td>On or after the earliest start date of the worker.</td>
<td>The effective end date of the contact relationship.</td>
</tr>
</tbody>
</table>
## Oracle Human Capital Management Cloud

### Integrating with HCM

#### Chapter 9

### Loading Workers

<table>
<thead>
<tr>
<th>Component</th>
<th>Earliest Effective Start Date</th>
<th>Last Effective End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Employment Terms</td>
<td>The first employment terms must have an effective start date equal to the start date of the corresponding work relationship. Subsequent employment terms can have an effective start date on or after the start date of the corresponding work relationship.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Assignment</td>
<td>The first assignment must have an effective start date equal to the earliest effective start date of any corresponding employment terms. Subsequent assignments can have an effective start date on or after the earliest effective start date of the corresponding employment terms. Further restrictions on the earliest effective start date may exist, depending on the employment model being used.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Assignment Extra Information</td>
<td>On or after the earliest effective start date of the corresponding assignment.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Assignment Grade Steps</td>
<td>On or after the earliest effective start date of the corresponding assignment.</td>
<td>The effective end date of the grade steps.</td>
</tr>
<tr>
<td>Assignment Manager</td>
<td>On or after the earliest effective start date of the corresponding assignment.</td>
<td>The effective end date of the manager relationship.</td>
</tr>
<tr>
<td>Assignment Work Measure</td>
<td>On or after the earliest effective start date of the corresponding assignment.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Contract</td>
<td>The earliest effective start date of the corresponding employment terms.</td>
<td>The end of time.</td>
</tr>
<tr>
<td>Worker Extra Information</td>
<td>On or after the earliest start date of the worker.</td>
<td>The end of time.</td>
</tr>
</tbody>
</table>

### Guidelines for Deleting Worker Components

You can delete some components of the Worker object but not the entire worker. For information about deletion support for individual Worker components, see the component details on the View Business Objects page. This topic provides additional information about deleting some of the Worker components using HCM Data Loader.

This table provides information about the circumstances in which you can delete some components of the Worker object.
<table>
<thead>
<tr>
<th>Component</th>
<th>Deletion Supported</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment Grade Steps</td>
<td>Yes</td>
<td>If you set the <code>GradedId</code> attribute of an assignment to <code>#NULL</code>, then any associated Assignment Grade Steps component is deleted automatically. Therefore, if you include a DELETE instruction for the Assignment Grade Steps component in the same .dat file as the assignment, an error occurs.</td>
</tr>
<tr>
<td>Person Address</td>
<td>Yes</td>
<td>Subject to country-specific regulations, you can delete any address. When multiple mailing addresses exist, you must delete all nonprimary mailing addresses before you can delete the primary mailing address. Alternatively, identify a new primary mailing address before you delete the current primary mailing address.</td>
</tr>
<tr>
<td>Person Email</td>
<td>Yes</td>
<td>When only one email exists, you can delete it. When multiple emails exist, you must delete all nonprimary emails before you can delete the primary email. Alternatively, identify a new primary email before you delete the current primary email.</td>
</tr>
<tr>
<td>Person Legislative Data</td>
<td>Yes</td>
<td>One legislative data record must exist for a worker. You can’t delete the only record. You can delete additional records.</td>
</tr>
<tr>
<td>Person Phone</td>
<td>Yes</td>
<td>When only one phone exists, you can delete it. When multiple phones exist, you must delete all nonprimary phones before you can delete the primary phone. Alternatively, identify a new primary phone before you delete the current primary phone.</td>
</tr>
<tr>
<td>User Information</td>
<td>No</td>
<td>The User Information component is available only when you create workers. This component isn’t available when you update workers.</td>
</tr>
<tr>
<td>Person User Manual Roles</td>
<td>No</td>
<td>The Person User Manual Roles component is available only when you create workers. This component isn’t available when you update workers. To remove a single role from a worker, you use the User object rather than the Worker object. To remove multiple roles from a worker, you use the User Role component of the User object.</td>
</tr>
<tr>
<td>Work Relationship</td>
<td>Yes</td>
<td>When deleting a work relationship, you must include the <code>CancelWorkRelationshipFlag</code> attribute with a value of <code>Y</code>.</td>
</tr>
</tbody>
</table>
How You Check for Duplicate Person Records

When you load person records using HCM Data Loader, you can request a check for duplicate records. In this case, an error message is raised if you try to load a duplicate record. This topic describes how to request duplicate checking for individual person records. It also describes how this option works in conjunction with the enterprise setting for duplicate checking.

Duplicate Checking for Individual Person Records

To request duplicate checking for a person record, you include the `PersonDuplicateCheck` attribute of the Worker object. This attribute can have one of the values shown in this table. The Description column identifies the attributes that the application uses in each case to identify duplicate records.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA_NONE or a blank value</td>
<td>No duplicate checking occurs</td>
</tr>
<tr>
<td>ORA_LN_FLIDOB_GEN_NID</td>
<td>Either the last name, first-name initial, date of birth, and gender or the national ID</td>
</tr>
<tr>
<td>ORA_LN_FLIDOB_NID</td>
<td>Either the last name, first-name initial, and date of birth or the national ID</td>
</tr>
<tr>
<td>ORA_LN_FNDOB_GEN_NID</td>
<td>Either the last name, first name, date of birth, and gender or the national ID</td>
</tr>
<tr>
<td>ORA_NID_ONLY</td>
<td>The national ID</td>
</tr>
</tbody>
</table>

For example, if you set `PersonDuplicateCheck` to `ORA_LN_FLIDOB_NID`, then a duplicate record is identified if one of these situations occurs:

- The last name, first-name initial, and date of birth all match those of an existing person record.
- The national ID matches that of an existing person record.

> **Note:** National ID values must be formatted. For example, to load the US social security number **987-65-4322**, you must include the hyphens. Don’t specify the number as **987654322**. If you omit the formatting, which is country-specific, then duplicate person records aren’t found when the checking is based on national identifier.

Enterprise Duplicate Checking

The enterprise option `Person Creation Service Duplicate Check` controls whether checks for duplicate person records occur by default when you load person records in bulk. If you exclude the `PersonDuplicateCheck` attribute of the Worker object, then the current setting of `Person Creation Service Duplicate Check` applies. If you include the `PersonDuplicateCheck` attribute, then the current setting of `Person Creation Service Duplicate Check` is ignored for the relevant person record.
Person Numbers in Worker Objects

Each person record, regardless of person type or number of work relationships, has a unique person number. Person numbers can be either generated automatically or entered manually, depending on the setting of the enterprise Person Number Generation Method option. This topic describes ways of providing person numbers for uploaded Worker objects.

Automatically Generated Person Numbers

To generate person numbers automatically, you set the Person Number Generation Method option to either Automatic Prior to Submission or Automatic Upon Final Save. When you load person records to environments where person numbers are generated automatically, you supply no person number. A number is generated automatically on upload.

When you load a person record without a person number, you must supply a source key to identify the person record uniquely. You must use the same key for all child components of the Worker object using the PersonId(SourceSystemId) attribute and hint.

Manually Entered Person Numbers

When the Person Number Generation Method option is set to Manual, you must supply a person number in the Worker object.

Loading Legacy Numbers

You can load legacy numbers to environments where person numbers are generated automatically. Set the Initial Person Number enterprise option to the highest legacy person number plus one so that the legacy sequence continues.

Correcting Person Numbers

You can correct a person number, provided that Person Number Generation Method is set to Manual.

To correct a person number when the enterprise method is automatic generation, you can:

1. Note which of the two automatic methods, Automatic Prior to Submission and Automatic Upon Final Save, is being used.

Note: You're recommended not to hire workers while the correction is being made.

3. Correct the person number.
4. Reinstall the original automatic number-generation method. Consider also whether the value of the Initial Person Number option must be updated to avoid conflict with the correction you made.
5. Run the Update Person Search Keywords process to ensure that person searches with person number as a keyword value are successful.

When you correct a person number, the correction applies to every date-effective update for the lifetime of the person record. This rule ensures that a person isn't identified using different person numbers at different times.

To identify the person record to correct, supply the source key, Oracle Fusion GUID, or Oracle Fusion surrogate ID. You can't supply the user key alone, as the person number is the user key.
Guidelines for Loading External Identifiers

The External Identifier component of the Worker object holds an identifier used by a third-party application, such as a time device or payroll application. This topic describes how to set some attributes of the External Identifier component.

External Identifiers at Person or Assignment Level

The external identifier can be captured:

- At the person level, by supplying only the person number
- At the assignment level, by supplying both the person number and the assignment number

External Identifier Types

The external identifier type is defined in lookup type ORA_PER_EXT_IDENTIFER_TYPES. You can add lookup codes to this lookup type.

Date Values

The DateFrom and DateTo attributes include a time stamp. The time stamp ensures that you can assign an external identifier of a single type to a worker more than once on any day. For example, a person may be assigned a time-device badge identifier that he or she loses the same day. If you replace the badge identifier on the same day, then the time stamps differentiate the external identifiers.

External Identifier Sequence

The ExternalIdentifierSequence attribute is a required component of the user key and must be unique. You're recommended to set this attribute to 1 for the first record for a worker. Increment it by 1 for subsequent records.

External Identifier Example

This example Worker.dat file loads an external identifier for a person.

```plaintext
SET PURGE_FUTURE_CHANGES N
METADATA|ExternalIdentifier|ExternalIdentifierNumber|PersonNumber|ExternalIdentifierType|
|ExternalIdentifierSequence
MERGE|ExternalIdentifier|rtyui45678|TestPerOTALTEST_9|Third-Party Payroll ID|1
```

Guidelines for Loading Person Names

The Person Name component of the Worker object holds both common and legislation-specific components of a person’s name. This topic describes some decisions you must make when loading person names using HCM Data Loader.
Local and Global Names

Person names are always created with both a global and a local version. The name is created based on the name type that you supply, and the other name is derived automatically. For example, if you set the NameType attribute to GLOBAL, then the local name is derived. Alternatively, if you set NameType to FR, for example, then the global name is derived. Typically, only the global name is required and the application copies it to the local name automatically. However, if the person’s name must be held in two different character sets, then supply the local name.

A core name field may not be available for a legislation. For example, Last Name Kanji isn’t available for the Japanese legislation. In this case, the data is stored in one of the NameInformation1 through NameInformation14 attributes. NameInformation1 through NameInformation14 are reserved for this purpose. The person name styles table stores the mapping of the name style to the name attributes. The name style is the same for both the global and local-name records.

Name Attributes for Specific Legislations

To supply a name attribute for a specific legislation, use one of the NameInformation15 through NameInformation30 attributes. For example, to define the name attribute Doing Business As for the United States, you could use the NameInformation15 attribute. You don’t have to define flexfields for this requirement.

Supplying Date-Effective History

If you’re loading date-effective history for a person name, then you must use source keys to identify the component. Otherwise, you can use the PersonNumber, EffectiveStartDate, and NameType user-key components.

Related Topics

- Oracle Fusion HCM: Person Name Validation (2146270.1)

Guidelines for Loading Person Images

The person image appears on various Oracle HCM Cloud application pages to identify a person’s record. This topic describes aspects of loading the Person Image component of the Worker object using HCM Data Loader.

Image Type

You can set the ImageType attribute only to PROFILE. If you omit the ImageType attribute, then the type is PROFILE by default.

Image Size

The recommended size of the image is 90 pixels by 120 pixels. Images of different sizes are supported, but you’re recommended to maintain an aspect ratio of 3 by 4 to reduce any distortion. No recommendation exists for the image resolution. The maximum size of the image file is 2 GB. However, you’re recommended to store images at the smallest possible size, taking into account the suggested dimensions. The smaller the image, the better the performance of the image display in the application. Typically, the file size of an image of the recommended dimensions would be only 2 or 3 MB, even at high resolution.
Examples of Loading Work Relationship Changes

The Worker object provides several indicator attributes that simplify maintenance of work relationships. Use these indicators to request actions such as termination or a change of primary assignment. This topic provides examples of work-relationship changes that you request using these indicators.

Note: These indicators aren’t supported when you’re supplying full date-effective history. They’re provided solely for independent actions. Also, you can’t provide multiple indicators on the same work relationship record. For example, you can’t both terminate a work relationship (TerminateWorkRelationshipFlag) and perform a global transfer (GlobalTransferFlag) on the same work relationship at the same time.

Changing the Hire Date

The hire date is the start date of a work relationship. To change the hire date, provide the new date on the NewStartDate attribute of the Work Relationship component. You can load just the work relationship record. The employment terms and assignment records are maintained automatically.

This example changes the hire date for an existing worker.

```
SET PURGE_FUTURE_CHANGES N
METADATA|WorkRelationship|SourceSystemOwner|SourceSystemId|PersonId(SourceSystemId)|LegalEmployerName|NewStartDate
MERGE|WorkRelationship|VISION|1009_POS|1009|Cox-6-CWB|2002/02/10
```

Terminating a Work Relationship

To terminate a work relationship, load the Work Relationship component with the TerminateWorkRelationshipFlag attribute set to Y. Specify the termination date on the ActualTerminationDate attribute. All associated assignments are automatically made inactive after the termination date.

This example terminates a work relationship.

```
SET PURGE_FUTURE_CHANGES N
METADATA|WorkRelationship|SourceSystemOwner|SourceSystemId|PersonId(SourceSystemId)|LegalEmployerName|TerminateWorkRelationshipFlag|ActualTerminationDate|ActionCode|ReasonCode
MERGE|WorkRelationship|VISION|1009_POS|1009|Cox-6-CWB|Y|2015/10/02|RESIGNATION|RESIGN_PERSONAL
```

Correcting the Termination Date of a Work Relationship

To correct a work relationship termination date, supply a new ActualTerminationDate and set the CorrectTerminationFlag attribute to Y.

This example corrects the termination date for a terminated work relationship.

```
SET PURGE_FUTURE_CHANGES N
METADATA|WorkRelationship|SourceSystemOwner|SourceSystemId|PersonId(SourceSystemId)|LegalEmployerName|CorrectTerminationFlag|ActualTerminationDate
```
Reversing a Termination

To reverse a termination, load the Work Relationship component with the `ReverseTerminationFlag` attribute set to `Y`.

This example reverses the termination of a work relationship.

```
SET PURGE_FUTURE_CHANGES N
METADATA|WorkRelationship|SourceSystemOwner|SourceSystemId|PersonId(SourceSystemId)|LegalEmployerName|
ReverseTerminationFlag
MERGE|WorkRelationship|VISION|1009_POS|1009|Cox-6-CWB|Y
```

Canceling a Work Relationship

To cancel a work relationship, use a DELETE instruction for the work relationship and set the `CancelWorkRelationshipFlag` attribute to `Y`.

This example deletes the specified work relationship and all its child records.

```
SET PURGE_FUTURE_CHANGES N
METADATA|WorkRelationship|SourceSystemOwner|SourceSystemId|PersonId(SourceSystemId)|CancelWorkRelationshipFlag
DELETE|WorkRelationship|VISION|1008_POS|1008|Y
```

Changing the Primary Assignment or Work Relationship

When you load a primary work relationship, the existing primary work relationship automatically becomes nonprimary on the date that you supply on the `DateForPrimaryFlagChange` attribute.

To make a nonprimary work relationship primary, you load the work relationship to be made primary with the `PrimaryFlag` attribute set to `Y`. Specify the date when the work relationship becomes primary on the `DateForPrimaryFlagChange` attribute. The corresponding changes to employment terms and assignments occur automatically. The primary indicator for the existing primary work relationship is set to `N` automatically.

How You Perform a Global Transfer

This topic explains how to perform global transfers for both primary and nonprimary work relationships using HCM Data Loader. A global transfer is a permanent transfer from one legal employer to another. It results in the termination of an existing work relationship and the creation of a new one. The termination date of the existing work relationship is one day prior to the start date of the new work relationship.
Performing a Global Transfer on a Primary Work Relationship

By default, the global transfer is applied to the primary work relationship. To perform a global transfer, you simply load the new Work Relationship component with all its child components. The new work relationship must have:

- A `GlobalTransferFlag` attribute set to `Y`
- An action code from the `EMPL_GLB_TRANSFER` action type
- A start date value that’s the date of the global transfer

Don’t include the work relationship that you’re terminating. That work relationship is terminated automatically.

The termination and creation of the work relationships are validated to ensure data consistency. In particular, a worker can’t be left with only an active, nonprimary work relationship. Therefore, if the worker has two active work relationships, you can’t perform the global transfer on the primary one. You must make the primary work relationship nonprimary before you attempt the global transfer.

Performing a Global Transfer on a Nonprimary Work Relationship

A worker can have multiple nonprimary work relationships. Therefore, when you perform a global transfer of a nonprimary work relationship, you must supply additional information so that the correct work relationship is terminated. To perform the global transfer on a nonprimary work relationship, you must also supply:

- The parent Worker component.
- The nonprimary work relationship that you’re terminating. However, don’t include:
  - Any child records for the terminated work relationship
  - The `GlobalTransferFlag` attribute

These records are in addition to those that you supply for a global transfer of a primary work relationship.

Updating Person-Name Legislation Codes

When you perform a global transfer, the new work relationship may have a legislation code that’s new for the person. In this case, you may want to update the legislation code for the person name. The person-name legislation code is validated against all available work-relationship legislation codes. Therefore, the change to the work relationship must occur before the change to the person name. You can update the legislation code in the person name in one of these ways:

- Include the Worker object with no change in the same .dat file as the changes to the Person Name and Work Relationship components. In this case, changes to the work relationship are processed before changes to the person name.
- Process updates to the work relationship before those to the person name.

Guidelines for Loading Additional Assignments

Whether you can load additional assignments for a worker depends on the employment model. This topic identifies, for each employment model, whether you can perform the Add Assignment action using HCM Data Loader.
### Employment Model vs. Add Assignment

<table>
<thead>
<tr>
<th>Employment Model</th>
<th>Add Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Tier - Single Assignment</td>
<td>No</td>
</tr>
<tr>
<td>2 Tier - Multiple Assignment</td>
<td>Yes</td>
</tr>
<tr>
<td>2 Tier - Single Contract - Single Assignment</td>
<td>No</td>
</tr>
<tr>
<td>2 Tier - Multiple Contract - Single Assignment</td>
<td>Yes</td>
</tr>
<tr>
<td>3 Tier - Single Employment Terms - Single Assignment</td>
<td>No</td>
</tr>
<tr>
<td>3 Tier - Single Employment Terms - Multiple Assignment</td>
<td>Yes</td>
</tr>
<tr>
<td>3 Tier - Multiple Employment Terms - Single Assignment</td>
<td>No</td>
</tr>
<tr>
<td>3 Tier - Multiple Employment Terms - Multiple Assignment</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The following example shows how to load an additional assignment. Both the Employment Terms and Assignment components are identified by user keys.

```plaintext
METADATA|WorkTerms|ActionCode|AssignmentNumber|AssignmentStatusTypeCode|AssignmentType|BusinessUnitShortCode|DateStart|EffectiveEndDate|EffectiveLatestChange|EffectiveSequence|EffectiveStartDate|LegalEmployerName|PersonNumber|PrimaryWorkTermsFlag|WorkerType
MERGE|WorkTerms|ADD_ASSIGN|ETTESTASSG03_562|ACTIVE_NO_PROCESS|ET|ZHRX-AE-Starter-BU|2016/01/01|4712/12/31|Y|2|2017/08/01|CRFL RRF LE 2T|TESTASSG03_56|Y|E

METADATA|Assignment|ActionCode|AssignmentNumber|AssignmentStatusTypeCode|AssignmentType|BusinessUnitShortCode|EffectiveEndDate|EffectiveLatestChange|EffectiveSequence|EffectiveStartDate|Frequency|LegalEmployerName|PersonNumber|PrimaryAssignmentFlag|WorkTermsNumber|WorkerType
MERGE|Assignment|ADD_ASSIGN|ETTESTASSG03_562|ACTIVE_NO_PROCESS|E|ZHRX-AE-Starter-BU|4712/12/31|Y|2|2017/08/01|W|CRFL RRF LE 2T|TESTASSG03_56|Y|ETTESTASSG03_562|E
```

### Guidelines for Loading Temporary Assignments

This topic describes restrictions that apply to the loading of *temporary assignments* using HCM Data Loader.

When you create a temporary assignment in the UI, two data rows are created:

1. One row is created for the temporary assignment.
2. One row is added to the base assignment, which is suspended.

Both rows have the action code TEMP_ASG and the same action occurrence ID, which is defined by the application.

When you end the temporary assignment, two further data rows are created:

1. One row is created to end the temporary assignment.
2. One row is added to the base assignment, which becomes active on the day following the end date of the temporary assignment.

Both rows have the action code `END_TEMP_ASG` and the same action occurrence ID.

Although you can create temporary assignments using HCM Data Loader, you can’t end them using HCM Data Loader. This restriction exists because no row is added to the base assignment when you load the temporary assignment. Therefore, no matching row is found in the base assignment when you attempt to end the temporary assignment.

The following example shows how to load a temporary assignment. Both the Employment Terms and Assignment components are identified by user keys.

```
METADATA|WorkTerms|ActionCode|AssignmentNumber|AssignmentStatusTypeCode|AssignmentType|BusinessUnitShortCode|DateStart|EffectiveEndDate|EffectiveLatestChange|EffectiveSequence|EffectiveStartDate|LegalEmployerName|PersonNumber|PrimaryWorkTermsFlag|WorkerType
MERGE|WorkTerms|TEMP_ASG|ETTESTASSG03_59|SUSPEND_PROCESS|ET|ZHRX-AE-Starter-BU|2016/01/01|4712/12/31|Y|1|2017/08/01|CRFL RRF LE 2T|TESTASSG03_59|Y|E
MERGE|WorkTerms|TEMP_ASG|ETTESTASSG03_59-2|ACTIVE_PROCESS|ET|ZHRX-AE-Starter-BU|2016/01/01|4712/12/31|Y|1|2017/08/01|CRFL RRF LE 2T|TESTASSG03_59|N|E
METADATA|Assignment|ActionCode|AssignmentNumber|AssignmentStatusTypeCode|AssignmentType|BusinessUnitShortCode|EffectiveEndDate|EffectiveLatestChange|EffectiveSequence|EffectiveStartDate|Frequency|LegalEmployerName|PersonNumber|PrimaryAssignmentFlag|WorkTermsNumber|WorkerType
MERGE|Assignment|TEMP_ASG|ETTESTASSG03_59|SUSPEND_PROCESS|E|ZHRX-AE-Starter-BU|4712/12/31|Y|1|2017/08/01|W|CRFL RRF LE 2T|TESTASSG03_59|Y|ETTESTASSG03_59|E
MERGE|Assignment|TEMP_ASG|ETTESTASSG03_59-2|ACTIVE_PROCESS|E|ZHRX-AE-Starter-BU|4712/12/31|Y|1|2017/08/01|W|CRFL RRF LE 2T|TESTASSG03_59|N|ETTESTASSG03_59-2|E
```

**Start Dates for Person Legislative Data**

This topic explains how to specify start dates for person legislative data that you load using HCM Data Loader.

Person legislative data must start on the person’s earliest **effective start date**. A person may have multiple **work relationships** in different legislations. In this case, all person legislative data must start on the person’s earliest effective start date, not the start date of the work relationship. When a person starts a work relationship that’s in a new legislation, the application:

- Creates a person legislative data record as of the person’s earliest effective start date with no data.
- Performs an update of the person legislative data record as of the date of the creation of the new work relationship with the user-provided data. This approach ensures that the data is chronologically correct.

For example, consider the work relationships of the worker in this table.

<table>
<thead>
<tr>
<th>Date</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>Hired by Vision Corporation in the UK</td>
</tr>
<tr>
<td>2005</td>
<td>Left Vision Corporation in the UK</td>
</tr>
<tr>
<td>2006</td>
<td>Joined US military</td>
</tr>
<tr>
<td>2009</td>
<td>Left US military and became a US veteran</td>
</tr>
<tr>
<td>2010</td>
<td>Hired by Vision Corporation in the US</td>
</tr>
</tbody>
</table>
In this example, in 2010 a US person legislative record is created as of 2000, the person’s earliest effective start date. The US legislative record is updated as of 2010, the start date of the person’s US work relationship. The update includes the descriptive flexfield details for US veteran status, as provided by the user.

Employment Terms Override at Assignment

This topic describes how to set attribute values on employment terms and their associated assignments when you load them using HCM Data Loader.

The enterprise option Allow Employment Terms Override at Assignment controls whether assignment attributes inherited from employment terms can be overridden on the assignment. However, when you load employment terms, the associated assignments don't inherit attribute values from the employment terms. This table summarizes how to set attribute values on employment terms to avoid conflicting with the current setting of Allow Employment Terms Override at Assignment.

<table>
<thead>
<tr>
<th>Allow Employment Terms Override at Assignment</th>
<th>Attribute Values on Employment Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Must match values on associated assignments</td>
</tr>
<tr>
<td>Yes</td>
<td>Can differ from values on associated assignments</td>
</tr>
</tbody>
</table>

Automatic Calculation of FTE Values for Workers

The full-time equivalent (FTE) value is the result of dividing assignment working hours by standard working hours, which are typically those of a full-time worker. For example, if the assignment working hours are 10 and the standard working hours are 40, then the FTE value is 0.25. When you edit assignment working hours in the user interface, FTE is calculated automatically. When you load assignment records using HCM Data Loader, you can specify FTE values on the Assignment Work Measure component. Alternatively, you can request that FTE values be calculated automatically. This topic describes how to request automatic calculation of FTE values.

Calculating FTE Values Automatically

To request automatic calculation of FTE values for all assignments in a Worker.dat file, include the following SET instruction in the file:

```
SET CALCULATE_FTE Y
```

When you include this instruction, the following rules apply:

- You must not include an Assignment Work Measure component for the assignment in the Worker.dat file.
- No more than one assignment work measure record can exist for the assignment in Oracle HCM Cloud.
- Both assignment working hours and standard working hours must exist for the assignment throughout the relevant time period. Otherwise, no calculation of FTE values occurs for the assignment.
You may find automatic calculation of FTE values helpful if:

- You're loading many assignment records that include assignment working hours.
- You aren’t loading FTE values in the Assignment Work Measure component.

By including the `SET CALCULATE_FTE Y` instruction in the .dat file, you can generate FTE values for all of the assignments.

⚠️ **Tip:** You can perform a data load with the sole purpose of calculating FTE values. Simply include the user keys of the assignments in a Worker.dat file that includes the `SET CALCULATE_FTE Y` instruction.

The `SET CALCULATE_FTE Y` instruction generates FTE values only. It doesn’t generate headcount values.

**Related Topics**
- File Line Instructions and File Discriminators
- The SET Instruction

---

Create a Default Working-Hour Pattern

A working-hour pattern defines the working hours, start time, and end time for each day of the week. When you load assignment records using HCM Data Loader, you can request that a default working-hour pattern record be created automatically.

**Creating a Default Working-Hour Pattern**

To create a default working-hour pattern for all assignments in a Worker.dat file, include this instruction in the Worker.dat file:

```
SET CREATE_DEFAULT_WORKING_HOUR_PATTERN Y
```

You may find automatic creation of default working-hour patterns helpful if you're loading many assignment records but aren’t loading the Working Hour Pattern component of the Worker object. Once this default record exists, you can update it with information for each worker by loading the Working Hour Pattern component of the Worker object.

**Related Topics**
- File Line Instructions and File Discriminators
- The SET Instruction

---

Examples of Loading Worker Working Hour Pattern

You can define shift types and working hours for your workers by loading working hour patterns with HCM Data Loader. If the daily working hours are a fixed number of hours per day, use Elapsed as the Work Shift Type to load. If you need to track the start and end times per day, use Time as the Work Shift Type to load. Let’s look at some Worker.dat files that create worker hour patterns.
Load Working Hour Pattern by Shift Type

To get started with working hour patterns, see the topic Create a Default Working-Hour Pattern. Then, you can use the following examples to load worker hours with the work shift type of your choice.

Let’s say you want to define the specific times of day in a worker’s daily shift. To do this, load the Working Hour Pattern using Time as the Work Shift Type.

METADATA|WorkTerms|ActionCode|AssignmentStatusTypeCode|AssignmentType|EffectiveEndDate|EffectiveLatestChange|EffectiveSequence|EffectiveStartDate|AssignmentNumber|LegalEmployerName|PrimaryWorkTermsFlag|DateStart|PersonNumber|WorkerType
MERGE|WorkTerms|HIRE|ACTIVE_PROCESS|ET|4712/12/31|Y|1|2019/03/29|E51323|V1_LE0227|Y|2019/03/13|51323|E
METADATA|Assignment|ActionCode|AssignmentStatusTypeCode|AssignmentType|EffectiveEndDate|EffectiveLatestChange|EffectiveSequence|EffectiveStartDate|WorkTermsNumber|AssignmentNumber|PrimaryFlag|PrimaryAssignmentFlag|HourlySalariedCode|PersonNumber
MERGE|Assignment|HIRE|ACTIVE_PROCESS|E|4712/12/31|Y|1|2019/03/29|ET966169008889921|E51323|Y|Salaried|51323
METADATA|WorkingHourPattern|Object|AssignmentNumber|EffectiveStartDate|WorkDayDefinitionCode|WorkingHoursType|WorkShiftType|SatEndTime|SatStartTime|FriStartTime|FriEndTime|MonStartTime|MonEndTime|SunEndTime|SunStartTime|ThuEndTime|ThuStartTime|TueEndTime|TueStartTime|WedStartTime|WedEndTime
MERGE|WorkingHourPattern|ASSIGNMENT|E51323|2019/03/29|TIME ENTRY STOP DAY|ORA_WORK_WEEK|ORA_TIME|18:00|10:00|10:00|18:00|10:00|18:00|10:00|18:00|10:00|10:00|18:00
If you would rather record how many hours worked per shift, you can set Work Shift Type as Elapsed.

Examples of Loading Assignment Eligible Jobs

For any worker assignment, you can identify additional jobs for which the worker is eligible. You can use this feature for workers who must report time for the additional jobs, for example. You can also use it simply to track the additional jobs of workers who have multiple jobs. This topic provides examples showing how to create and update Assignment Eligible Job components of the Worker object using HCM Data Loader.

Before you load Assignment Eligible Job components:

- The assignment job must exist in the target environment.
- The job family must exist in the target environment if the relief type is Derived.

Relief type can be either Manual or Derived.

- When the relief type is Derived, you can add jobs belonging to the job family of the worker’s assignment only. You can specify the ReliefType value using either the meaning Derived or the lookup code ORA_D.
When the relief type is **Manual**, you can add any job. You must specify **ManualRate** and **Frequency** values. You can specify the **ReliefType** value using either the meaning **Manual** or the lookup code **ORA_M**.

Also, the start date of the eligible job must not be before the assignment start date.

You load assignment eligible jobs data in the Worker.dat file for processing by HCM Data Loader. You can verify this data on the Manage Eligible Jobs page for a selected worker.

### Creating Assignment Eligible Jobs

This example Worker.dat file creates an assignment eligible job for a worker using source keys. **ReliefType** is **Manual**.

```
METADATA|AssignmentEligibleJob|AssignmentId(SourceSystemId)|JobId(SourceSystemId)|ToDate|FromDate|
SourceSystemOwner|SourceSystemId|ReliefType|ManualRate|Frequency
MERGE|AssignmentEligibleJob|1031101972|100000011571171|2000/04/01|2000/01/01|VISION|TEST_RC_ut0001|ORA_M|11.7|Hourly
```

This example Worker.dat file creates an assignment eligible job for a worker using source keys. **ReliefType** is **Derived**.

```
METADATA|AssignmentEligibleJob|AssignmentId(SourceSystemId)|JobId(SourceSystemId)|ToDate|FromDate|
SourceSystemOwner|SourceSystemId|ReliefType
MERGE|AssignmentEligibleJob|1031101860|100000011571171|2000/04/01|2000/01/01|VISION|TEST_RC_ut0002|ORA_D
```

This example Worker.dat file creates assignment eligible jobs for a worker using user keys. **ReliefType** is **Manual**. The user key attributes for this component are **AssignmentNumber**, **JobCode**, **BusinessUnitShortCode**, and **EffectiveStartDate**.

```
METADATA|AssignmentEligibleJob|AssignmentNumber|JobCode|ToDate|FromDate|BusinessUnitShortCode|ReliefType|ManualRate|Frequency
MERGE|AssignmentEligibleJob|TEST_ASG_ut0001|JOBOPMANCORE|2000/04/01|2000/01/01|Vision Corporation Enterprise|Manual|11.7|Hourly
```

### Updating Assignment Eligible Jobs

You can update only the value of the **ToDate** attribute of an assignment eligible job. This example Worker.dat file updates a worker’s assignment eligible job using source keys.

```
METADATA|AssignmentEligibleJob|AssignmentId(SourceSystemId)|JobCode|ToDate|FromDate|BusinessUnitShortCode|ReliefType|ManualRate|Frequency
SourceSystemOwner|SourceSystemId|ReliefType|ManualRate|Frequency
MERGE|AssignmentEligibleJob|1031101972|JOBOPMANCORE|2001/04/01|2000/01/01|Vision Corporation Enterprise|VISION|TEST_RC_ut0001|Manual|11.7|Hourly
```

This example Worker.dat file updates a worker’s assignment eligible job using user keys.

```
METADATA|AssignmentEligibleJob|AssignmentNumber|JobCode|ToDate|FromDate|BusinessUnitShortCode|ReliefType|ManualRate|Frequency
MERGE|AssignmentEligibleJob|TEST_ASG_ut0001|JOBOPMANCORE|2001/04/01|2000/01/01|Vision Corporation Enterprise|Manual|11.7|Hourly
```

### Guidelines for Loading Seniority Dates

A seniority date is the date on which the calculation of a worker’s length of service with the enterprise is based. A worker’s status, rank, or entitlement to benefits may depend on his or her seniority. The seniority date can be based on the worker’s
time in a particular entity, such as a legal employer or job. Therefore, you can configure seniority rules based on values including collective agreement, country, department, enterprise, grade, job, legal employer, location, position, and union. This topic describes some factors that affect how you load seniority dates using HCM Data Loader.

Versions of the Seniority Dates Functionality

Three versions of the seniority dates functionality, referred to as V1, V2, and V3, exist. The following table identifies:

- The user interface task that you use to manage each version
- Whether you can update the seniority dates for each version using HCM Data Loader

<table>
<thead>
<tr>
<th>Version</th>
<th>User Interface Task</th>
<th>Update Using HCM Data Loader</th>
<th>Worker Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>Manage Work Relation</td>
<td>Yes</td>
<td>Work Relationship</td>
</tr>
<tr>
<td>V2</td>
<td>Manage Seniority Dates, without fast formula support</td>
<td>No</td>
<td>Not applicable</td>
</tr>
<tr>
<td>V3</td>
<td>Manage Seniority Dates, with fast formula support</td>
<td>Yes</td>
<td>Seniority Date</td>
</tr>
</tbody>
</table>

Note: After you migrate to V3 seniority dates, you can’t use either V1 or V2. Therefore, you can’t load seniority dates on the Work Relationship component. This topic applies to V3 seniority dates.

For more information, see Comparison Between Different Seniority Dates Versions (2414630.1) on My Oracle Support at https://support.oracle.com.

Running the Calculate Seniority Dates Process

You can update but not create V3 seniority records for workers using HCM Data Loader. You must run the Calculate Seniority Dates process to create default seniority records for workers, based on configured seniority date rules, before you update them. After updating seniority records using HCM Data Loader, you can verify them for a worker on the Manage Seniority Dates page.

Required Attributes

The following table identifies attribute values that you must include in the Seniority Date component of the Worker object to associate it with a seniority date rule. These requirements depend on the level at which the seniority date rule is configured.

<table>
<thead>
<tr>
<th>Level of Seniority Date Rule</th>
<th>Required Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work relationship</td>
<td>One of:</td>
</tr>
<tr>
<td></td>
<td>· PeriodOfServiceld</td>
</tr>
</tbody>
</table>

Note: After you migrate to V3 seniority dates, you can’t use either V1 or V2. Therefore, you can’t load seniority dates on the Work Relationship component. This topic applies to V3 seniority dates.

For more information, see Comparison Between Different Seniority Dates Versions (2414630.1) on My Oracle Support at https://support.oracle.com.
Oracle Human Capital Management Cloud
Integrating with HCM

Chapter 9
Loading Workers

Level of Seniority Date Rule | Required Attributes
--- | ---
| • LegalEmployerName, DateStart, and WorkerType

Assignment

One of:
• AssignmentId
• AssignmentNumber

Person

One of:
• PersonId
• PersonNumber

Updating Seniority Dates Using Source Keys

As you can't create V3 seniority records using HCM Data Loader, they have default source keys where the source-system owner is FUSION. Therefore, to update seniority dates using source keys, you must:

• Extract their default source keys using the HCM extract Integration Object User Key Extract.
• Update them using the Source Key object so that all source-key references for a single Seniority Date component have the same source-system owner.

Related Topics
• Comparison Between Different Seniority Dates Versions

Examples of Loading Seniority Date Adjustments

A seniority date is the date on which the calculation of a worker’s length of service with the enterprise is based. This topic provides examples showing how to update V3 seniority dates using HCM Data Loader.

The following table identifies attributes that you can use to identify the entity on which the seniority date is based. You can use alternative key types, where available. For example, to identify a job you can use the user key (JobCode and BusinessUnitShortCode) instead of the JobId value.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Identifying Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bargaining unit</td>
<td>BargainingUnitCode</td>
</tr>
<tr>
<td>Collective agreement</td>
<td>CollectiveAgreementId</td>
</tr>
<tr>
<td>Department</td>
<td>DepartmentId</td>
</tr>
<tr>
<td>Enterprise</td>
<td>BusinessUnitShortCode</td>
</tr>
<tr>
<td>Grade</td>
<td>Gradeld</td>
</tr>
<tr>
<td>Grade step</td>
<td>GradeStepId</td>
</tr>
</tbody>
</table>
### Updating Seniority Dates Using User Keys

The following example Worker.dat files update a worker’s seniority date based on various assignment attributes. They use user keys to reference the Seniority Date component. As these examples show, you must include the ManualAdjustmentComments attribute when you make a manual adjustment to the seniority date. In these examples, the seniority date rule is configured at the person level.

This example updates a worker’s seniority date based on the assignment.

```
MERGE|SeniorityDate|JUN_A|9990101|2012/01/01|2015/01/01|4712/12/31|Vision Corporation Enterprise|20|Data Correction|JOBCD7|E9990401
```

This example updates a worker’s seniority date based on the department.

```
MERGE|SeniorityDate|JUN_DEPARTMENT|9990101|2012/01/01|2015/01/01|4712/12/31|Vision Corporation Enterprise|20|Data Correction|Commercial Sales
```

This example updates a worker’s seniority date based on the enterprise.

```
MERGE|SeniorityDate|JUN_ET|9990101|2012/01/01|2015/01/01|4712/12/31|Vision Corporation Enterprise|20|Data Correction
```

This example updates a worker’s seniority date based on the grade.

```
MERGE|SeniorityDate|JUN_GRADE|9990101|2012/01/01|2015/01/01|4712/12/31|Vision Corporation Enterprise|20|Data Correction|DHRX-AE-IC1
```

This example updates a worker’s seniority date based on the job.

```
MERGE|SeniorityDate|JUN_JOB|9990101|2012/01/01|2015/01/01|4712/12/31|Vision Corporation Enterprise|20|Data Correction|JOBCD7
```

---

<table>
<thead>
<tr>
<th>Entity</th>
<th>Identifying Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job</td>
<td>JobId</td>
</tr>
<tr>
<td>Legal employer</td>
<td>LegalEntityId</td>
</tr>
<tr>
<td>Legislation</td>
<td>LegislationCode</td>
</tr>
<tr>
<td>Location</td>
<td>LocationId</td>
</tr>
<tr>
<td>Position</td>
<td>PositionId</td>
</tr>
<tr>
<td>Union membership</td>
<td>UnionId</td>
</tr>
</tbody>
</table>
Examples of Loading Seniority Hours

You calculate seniority in hours for hourly paid workers. So that the application can perform seniority calculations for these workers, you load their seniority hours using HCM Data Loader. You can obtain the seniority hours from various sources, such as Oracle Time and Labor, Oracle Global Payroll, or a third-party application. Any worker assignment for which you load seniority hours must exist and be active as of the start date of the hours that you’re loading. It must also be paid hourly for the entire period for which you’re loading data.

Tip: You can verify the loaded data for a worker on the Manage Seniority Dates page. To verify loaded data for multiple workers, you can query the PER_SENIORITY_HOURS table.

This topic provides examples showing how to create, update, and delete the Seniority Hour component of the Worker object using HCM Data Loader.
Creating Seniority Hours

This example Worker.dat file shows how to create the Seniority Hour component of the Worker object using source keys.

```
METADATA|SeniorityHour|AssignmentId(SourceSystemId)|ToDate|FromDate|Hours|SourceSystemOwner|SourceSystemId
MERGE|SeniorityHour|1031101972|2001/01/01|2000/01/01|500|VISION|UT00214
```

This example Worker.dat file shows how to create the Seniority Hour component of the Worker object using user keys. The user key attributes for this component are `AssignmentNumber` and `FromDate`.

```
METADATA|SeniorityHour|AssignmentNumber|ToDate|FromDate|Hours
MERGE|SeniorityHour|E00214|2001/01/01|2000/01/01|500
```

Updating Seniority Hours

This example Worker.dat file shows how to update the Seniority Hour component of the Worker object using source keys.

```
METADATA|SeniorityHour|AssignmentId(SourceSystemId)|ToDate|FromDate|Hours|SourceSystemOwner|SourceSystemId
MERGE|SeniorityHour|1031101972|2001/01/01|2000/01/01|600|VISION|UT00214
```

This example Worker.dat file shows how to update the Seniority Hour component of the Worker object using user keys.

```
METADATA|SeniorityHour|AssignmentNumber|ToDate|FromDate|Hours
MERGE|SeniorityHour|E00214|2001/01/01|2000/01/01|600
```

Deleting Seniority Hours

This example Worker.dat file shows how to delete the Seniority Hour component of a Worker object using source keys.

```
METADATA|SeniorityHour|AssignmentId(SourceSystemId)|SourceSystemOwner|SourceSystemId
DELETE|SeniorityHour|1031101972|VISION|UT00214
```

This example Worker.dat file shows how to delete the Seniority Hour component of a Worker object using user keys.

```
METADATA|SeniorityHour|AssignmentNumber|ToDate|FromDate|Hours
DELETE|SeniorityHour|E00214|2001/01/01|2000/01/01|500
```

Example of Loading a Contract Extension

You supply the Contract component of the Worker object when creating workers, if appropriate. You can’t load Contract components later. However, you can load a contract extension for a worker who has a contract. This topic shows how to load a contract extension using HCM Data Loader.
Loading a Contract Extension

You can use HCM Data Loader to give an existing worker a contract extension. Keep these rules in mind:

- You must load the relevant Employment Terms, Assignment, and Contract components in the same Worker.dat file.
- The values of the **EffectiveStartDate** and **EffectiveEndDate** attributes must be the same on all three components.
- The **EffectiveStartDate** value is the start date of the extension period. This value must be the projected end date of the last contract period, plus one day, meaning no gaps are permitted between contract periods.
- The **EffectiveEndDate** value must be the end of time if this contract record is the latest record. It must not be the contract end date.
- You can specify the extension period by updating the **ExtensionPeriod** and **ExtensionPeriodUnits** attributes of the Contract component.
  
  In this case, the new projected end date is calculated automatically.
- The value of the **ActionCode** attribute on the Employment Terms and Assignment components must be associated with the action type EMPL_CONTRACT_EXTN.
- You can't delete a contract extension record using HCM Data Loader.

The following example shows how to load the Employment Terms, Assignment, and Contract components for a contract extension.

```
METADATA|WorkTerms|SourceSystemOwner|PersonId(SourceSystemId)|PeriodOfServiceId(SourceSystemId) |
SourceSystemId|AssignmentId|EffectiveStartDate|EffectiveEndDate|EffectiveSequence|EffectiveLatestChange|ActionCode
MERGE|WorkTerms|FUSION|300100162474267|300100162474475|300100162474482|300100162474482|2011/01/31|4712/12/31|1|Y|CONTRACT_EXTENSION
METADATA|Assignment|SourceSystemOwner|PersonId(SourceSystemId)|PeriodOfServiceId(SourceSystemId) |
WorkTermsAssignmentId(SourceSystemId)|SourceSystemId|AssignmentId|EffectiveStartDate|EffectiveEndDate|
EffectiveSequence|EffectiveLatestChange|ActionCode
MERGE|Assignment|FUSION|300100162474267|300100162474475|300100162474482|300100162474482|2011/01/31|4712/12/31|1|Y|CONTRACT_EXTENSION
METADATA|Contract|SourceSystemOwner|PersonId(SourceSystemId)|PersonNumber|AssignmentId(SourceSystemId) |
SourceSystemId|ContractId|EffectiveStartDate|EffectiveEndDate|ExtensionPeriod|ExtensionPeriodUnits
MERGE|Contract|FUSION|300100162474267|9600000000000615|300100162474482|300100162474490|300100162474490|2011/01/31|4712/12/31|100|D
```

Example of Loading a Default Expense Account for a Worker

For every worker who uses Oracle Expenses Cloud, a default expense account must be defined. The Default Expense Account **key flexfield** is available on the Assignment component of the Worker object. This topic shows how to load the default expense account for a worker on the worker **assignment**.
Loading the Default Expense Account on the Assignment

If the expense account is configured with multiple segments, then provide the concatenated value. Separate the segment values using the separator configured for the expense account, as shown in this example.

```
MERGE|Assignment|E10020592|2012/08/13|2015/04/30|1|Y|MIGRATED|3000000001934525|5005 - ICT|GB|NICT Manager|M|
5101.0000.3030.0000.231200.0000.00000000.0000.0000|10020592_WORKTERMS|VISION|10020592_ASSIGNMENT

MERGE|Assignment|E10020592|2015/05/01|4712/12/31|1|Y|MIGRATED|3000000001934525|5005 - ICT|GB|NICT Manager|M|
5101.0000.3030.0000.231200.0000.00000000.0000.0000|10020592_WORKTERMS|VISION|10020592_ASSIGNMENT
```

FAQs for Loading Workers

How do I load person phones?

You must include any formatting characters in the value that you specify on the **PhoneNumber** attribute of the Person Phone component. For example, to enter the US phone 650.555.0185 in the UI, you enter 650 in the **Area Code** field and 555-0185 in the **Number** field. When loading this number using HCM Data Loader, you must specify the value 555-0185 on the **PhoneNumber** attribute.

Formatting requirements vary by legislation. Always include the formatting characters, such as hyphen or period, specified for your legislation when loading person phones using HCM Data Loader.

If you include the **LegislationCode** attribute, then any **CountryCodeNumber** value that you supply must be valid for the specified legislation code.

How do I load person national identifiers?

You must include any formatting characters in the value that you specify on the **NationalIdentifierNumber** attribute of the Person National Identifier component. For example, to load the US social security number 987-65-4322, you must include the hyphens. Don't specify the number as 987654322. If you omit the formatting, which is country-specific, then duplicate person records aren’t found when the checking is based on national identifier.

How do I specify probation values on an assignment?

You can supply either the **ProbationPeriod** and **ProbationUnit** values or the **DateProbationEnd** value.

- If you supply the **ProbationPeriod** and **ProbationUnit**, then the **DateProbationEnd** value is calculated automatically.
- If you supply the **DateProbationEnd**, then the **ProbationPeriod** and **ProbationUnit** values are calculated automatically.
- If you supply values for all three attributes, then an error is raised.
Until update 17D, if you supplied the \texttt{DateProbationEnd}, then the \texttt{ProbationPeriod} and \texttt{ProbationUnit} values weren’t calculated automatically. In addition, if you supplied all three attributes, then the \texttt{DateProbationEnd} value was calculated automatically and the values you supplied were discarded. These errors are now fixed.
10 Loading Worker-Related Objects

Example of Loading an Allocated Checklist

An allocated checklist is a specific instance of a checklist template that's assigned to one or more workers. The checklist contains tasks that workers must perform. This topic shows how to load an allocated checklist using HCM Data Loader.

Creating an Allocated Checklist

This example AllocateChecklist.dat file creates two allocated checklists for the same person. Contacts and Contents components are created for Checklist 1, and a Tasks component with a notification request is created for Checklist 2. All components are referenced by user keys.

```
COMMENT Create two allocated checklists
METADATA|AllocateChecklist|ChecklistName|Country|ChecklistCategory|ChecklistInstance|PersonNumber|AllowAutoAllocation
MERGE|AllocateChecklist|Checklist_1|US|ORA_ONB_ENT_ONBOARDING|1|8153818|N
MERGE|AllocateChecklist|Checklist_2|US|ONBOARD|1|8153818|N

COMMENT Create checklist contacts
METADATA|AllocateChecklistContacts|ContactTitleCode|ContactType|PersonNumber|Country|ChecklistName|ChecklistCategory|ChecklistInstance
MERGE|AllocateChecklistContacts|ORA_ONB_ONBOARDING_SPONSOR|ORA_ONB_LN_MGR|8153818|US|Checklist_1|ORA_ONB_ENT_ONBOARDING|1

COMMENT Create checklist contents
METADATA|AllocateChecklistContents|ContentType|ChecklistName|ChecklistCategory|ChecklistInstance|PersonNumber|ContentDefnCode|Country
MERGE|AllocateChecklistContents|ORA_ONB_EVENT|Checklist_1|ORA_ONB_ENT_ONBOARDING|1|8153818|CHKLIST_CONT_DEF67|US

COMMENT Create allocated checklist tasks
METADATA|AllocateChecklistTask|TaskName|ChecklistName|Country|ChecklistCategory|ChecklistInstance|PersonNumber|MandatoryFlag|TaskPerformerPersonNumber|TaskOwnerPersonNumber
MERGE|AllocateChecklistTask|TaskChecklist_2|Checklist_2|US|ONBOARD|1|8153818|Y|8153818|8153818

COMMENT Create allocated checklist task notifications
METADATA|AllocateChecklistTaskNotifications|NotifyOwner|NotifyPerformer|TaskEvent|ChecklistName|ChecklistCategory|ChecklistInstance|Country|PersonNumber|TaskName
MERGE|AllocateChecklistTaskNotifications|Y|Y|ORA_CHK_TASK_ASSIGN|Checklist_2|ONBOARD|1|US|8153818|TaskChecklist_2
```

These rules apply when you load allocated checklists using HCM Data Loader:

- Eligibility profiles for checklists and tasks aren’t evaluated.
- Workflow notifications aren’t triggered for the checklist tasks.
- The allocation happens immediately, regardless of any allocation date specified in the .dat file.

Related Topics

- Guidelines for Loading Checklist Templates
Guidelines for Loading Areas of Responsibility

Areas of responsibility are responsibilities that a worker has as part of his or her job. Such responsibilities have a defined scope, such as a country or department. For example, a worker may be:

- The Human Resources Representative for a group of organizations
- The Benefits Representative for workers with a specified job
- The Union Representative for workers in Germany

Areas of responsibility are a recommended way of securing access to person records. This topic describes how to load areas of responsibility for a person using HCM Data Loader. Use the AreasOfResponsibility.dat file.

Responsibility Type
Responsibility types must exist in the target environment before you can load areas of responsibility. These five responsibility types are available by default, but you can add your own to suit business needs:

- Benefits Representative
- Human Resources Representative
- Payroll Representative
- Recruiting
- Union Representative

The responsibility type is validated against the lookup type PER_RESPONSIBILITY_TYPES.

Responsibility Name
The responsibility name must be unique for the combination of person and responsibility type.

Scope of Responsibility
Various attributes available under scope of responsibility, such as business unit, legal employer, department, and location, must exist before you can load areas of responsibility. For example, a location must exist before you can identify it as the scope of an area of responsibility.

Deleting Areas of Responsibility
You can delete areas of responsibility using HCM Data Loader. However, ensure that deleting an area of responsibility doesn’t have unintended consequences. For example, a worker’s secured access to person records may be lost if you delete the area of responsibility on which it’s based.

Related Topics
- How You Assign Areas of Responsibility
- Guidelines for Securing Person Records
Guidelines for Loading Document Records

Document records store information about documents, such as visas, licenses, and medical certificates, for a person. Document records can include electronic versions of the documents as attachments. This topic describes how to load Document Record and Document Record Attachment components for a person using HCM Data Loader.

Document Type

Document records exist for a specific document type. You must ensure that the document type exists in the target environment. The definition of the document type identifies supported and required attributes.

Attachment Files

The Document Record Attachment component holds the electronic version of a document for a person. Use the attribute URLorTextorFilename to supply the:

- **URL**, if the DataTypeCode is WEB_PAGE
- **Text**, if the DataTypeCode is TEXT
- **File name**, if the DataTypeCode is FILE

If you’re uploading attachment files, then they must be in the same .zip file as the DocumentsOfRecord.dat file that contains the related document records. All attachment files must be placed in a subdirectory named BlobFiles. This subdirectory is named for the BLOB data-type format in which attachment files are held. You specify the name of the attachment file on the File attribute of the Document Record Attachment component.

Loading Document Records

You supply document record data in a DocumentsOfRecord.dat file. This example DocumentsOfRecord.dat file creates a passport document for a person and includes a PDF file of the passport as an attachment. It identifies both the document record and the attachment using source keys.

```
METADATA|DocumentsOfRecord|SourceSystemOwner|SourceSystemId|PersonId(SourceSystemId)|DocumentCode|DocumentName|DateFrom|DateTo|IssuingAuthority|IssuingCountry
MERGE|DocumentsOfRecord|VISION|56883|134003891|PASSPORT_2442|UK Passport|2012/04/03|2022/04/03|United Kingdom
Passport Control
METADATA|DocumentAttachment|SourceSystemOwner|SourceSystemId|DocumentsOfRecordId(SourceSystemId)|DataTypeCode|URLorTextorFilename|File
MERGE|DocumentAttachment|VISION|90982|56883|FILE|Passport_2442|JSmithPassport.pdf
```

Deleting Document Records

You can delete Document Record objects using HCM Data Loader. When you delete a document record, any associated attachment record is deleted automatically. You can delete just the attachment, if appropriate.

This example DocumentsOfRecord.dat file deletes an existing document record. It identifies the document record using its source key.

```
METADATA|DocumentsOfRecord|SourceSystemOwner|SourceSystemId
DELETE|DocumentsOfRecord|VISION|56883
```
Related Topics

- How You Load Images, Attachments, and Large Strings

Guidelines for Loading Time Record Groups

A time record group includes related time entries, such as all reported time entries in a weekly time card. You can use HCM Data Loader to load only approved time record groups, that is, historical data. You can’t edit time record groups after upload. This topic describes aspects of the Time Record Group object that you must understand to load time record groups successfully using HCM Data Loader.

Preparing to Load Time Record Groups

Before you can load time record groups:

- Setup of Oracle Fusion Time and Labor must be complete.
- Relevant person records, each with a valid worker assignment, must exist in the target environment.
- A worker time processing profile must exist for each worker whose time card details you’re loading. The time processing profile may identify other consumers of the time card data.
  - If Oracle Fusion Global Payroll is a consumer of time card data, then the relevant payroll element entries must exist.
  - If Oracle Fusion Project Costing is a consumer of time card data, then the relevant Project Costing setup data must exist.

You may also need:

- Time rules, if the data being loaded required validation and calculation
- A worker time entry profile, if the worker or time and labor manager can view the loaded data in the application

Supplying Keys

The Time Record Group object doesn’t support source keys. Therefore, supply user keys when creating time record groups.

Querying Predefined Time Attributes

Predefined time attributes and their data types aren’t visible on a user interface. To query time attribute names, use this SQL query:

```sql
SELECT ATRB.NAME, ATRB.DISPLAY_NAME, ATRB.CLASS, ATRB.ATTRIBUTE_TYPE
FROM HWM_TM_ATRB_FLDS_VL ATRB
WHERE ATRB.ATTRIBUTE_CATEGORY NOT IN ('TIME_BUILDING_BLOCK', 'TIME_RECORD', 'TIME_RECORD_GROUP');
```

This table shows example results from this query.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Attribute Display Name</th>
<th>Attribute Class</th>
<th>Attribute Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>PJC_PROJECT_ID</td>
<td>Project</td>
<td>SIMPLE</td>
<td>Number</td>
</tr>
</tbody>
</table>
Updating Time Record Groups

When updating a time record group, you must supply the complete Time Record Group object in the TimeRecordGroup.dat file, not just the changed records.

Deleting Time Record Groups

You can delete Time Record Group objects using HCM Data Loader only if you identify them using surrogate IDs. When you delete a Time Record Group object, its child components are also deleted. You can’t delete individual components of the object. This example TimeRecordGroup.dat file deletes a time record group and its child components. It references the record by its surrogate ID.

METADATA|TimeRecordGroup|TimeRecordGroupId|ResourceId|ResourceType|TcStartTime|TcStopTime|GroupType|SubResourceId
DELETE|TimeRecordGroup|100000001|100000008153756|PERSON|2010/01/04 00:00:01|2010/01/09 23:59:59|RPTD_TIME|100000008154060

To obtain the surrogate IDs of existing objects, use the HCM Data Loader extract Integration Object User Key Map Extract.

Related Topics

• HCM Data Loader Extracts

Examples of Loading Time Record Groups

This topic shows how to create Time Record Group objects using HCM Data Loader.

Creating Time Record Groups

This example TimeRecordGroup.dat file creates two time record groups for the same assignment. It references the person and assignment by their user keys.

METADATA|TimeRecordGroup|GroupType|ResourceType|PersonNumber|AssignmentNumber|TcStartTime|TcStopTime
MERGE|TimeRecordGroup|RPTD_TIME|PERSON|8153756|EEEE8153756|2010/02/01 00:00:01|2010/02/06 23:59:59
METADATA|TimeRecord|GroupType|ResourceType|PersonNumber|TcStartTime|TcStopTime|OrderEntered|StartTime|StopTime|TmRecType|AssignmentNumber|Measure|UnitOfMeasure
MERGE|TimeRecord|RPTD_TIME|PERSON|8153756|2010/02/01 00:00:01|2010/02/06 23:59:59|1|2010/02/01 09:00:01|2010/02/01 13:00:01|RANGE|EEEE8153756|4|HR
MERGE|TimeRecord|RPTD_TIME|PERSON|8153756|2010/02/02 00:00:01|2010/02/06 23:59:59|2|2010/02/02 09:00:01|2010/02/02 13:00:01|RANGE|EEEE8153756|4|HR
MERGE|TimeRecord|RPTD_TIME|PERSON|8153756|2010/02/03 00:00:01|2010/02/06 23:59:59|3|2010/02/03 09:00:01|2010/02/03 13:00:01|RANGE|EEEE8153756|4|HR
MERGE|TimeRecord|RPTD_TIME|PERSON|8153756|2010/02/04 00:00:01|2010/02/06 23:59:59|4|2010/02/04 09:00:01|2010/02/04 13:00:01|RANGE|EEEE8153756|4|HR
MERGE|TimeRecord|RPTD_TIME|PERSON|8153756|2010/02/05 00:00:01|2010/02/06 23:59:59|5|2010/02/05 09:00:01|2010/02/05 13:00:01|RANGE|EEEE8153756|4|HR
This example demonstrates that:

- The **GroupType** attribute value on all components must be **RPTD_TIME**.
- The **ResourceType** attribute value on all components must be **PERSON**.
- The **TcStartTime** and **TcStopTime** attribute values on all child components of the object must match those on the parent Time Record Group object.
- You must supply the same value for the **OrderEntered** attribute on both the Time Repository Attribute component and its parent Time Record component.

### Guidelines for Loading User-Update Requests

You use the User object to create requests to update existing Oracle HCM Cloud user accounts. This topic describes when to use each component of the User object. It also describes any restrictions on its use.

#### The User Component

You can use the User component to create requests to:

- Update a user name.
- Suspend or activate a user.
- Add a single role to or remove a single role from a user.
- Create a user account, in specific circumstances.
- Delete a user account.

You also use this component to update the **CredentialsEmailSent** value in the PER_USERS table. This value records whether the account and password details have been sent to the user. Valid values are **Y** and **N**. After the email has been sent, it can't be sent again unless you reset this value. For example, to reset user passwords, you can run the **Send User**
**Name and Password Email Notifications** process. Before doing so, you can set CredentialsEmailSent to N to ensure that users are notified of new passwords. Otherwise, they aren’t notified.

The User Role Component

The User Role component is a child of the User object. You use the User Role component to add or remove multiple roles for a user. To add or remove a single role, use the User component.

Creating User Accounts

Depending on the setting of the **User Account Creation** enterprise option, user accounts can be created automatically for workers loaded in bulk. However, you may have prevented the user account from being created for an individual worker. Alternatively, it may have failed to be created for some reason. In these cases, you can use the User object to request a user account.

> **Note:** Use the User object to request a user account only as an exception. The standard method is to request the user account when you create the worker.

A user account can be created by means of the User object only if the **User Account Creation** enterprise option enables automatic creation of user accounts.

Deleting User Accounts

On occasion, you may want to delete a user account. For example, you may have created the user name in the wrong format. You can delete user accounts in both test and production environments, and the accounts can be in any status. The User and Role Provisioning options have no effect on user-account deletion. For example, deletion requests can’t be suppressed.

> **Tip:** The user is deleted from both the LDAP directory and the PER_USERS, PER_USER_ROLES, and PER_USER_HISTORY records. Therefore, you can reuse the user name. You can recreate the user account on the Security Console, using the **Manage Users** task, or by loading the User object.

Running Send Pending LDAP Requests

After loading user-update requests, you run the **Send Pending LDAP Requests** process to deliver them to your LDAP directory server. The recommendation is to schedule this process to run daily. You don’t have to run this process after updating the CredentialsEmailSent indicator.

**Related Topics**

- Why You Run the Send Pending LDAP Requests Process

Examples of Loading User-Update Requests

Use the User object to manage existing users and their roles. This topic provides examples showing how to create some typical user-update requests.
Updating a User

This example creates a request to update the user name for the user identified by the specified `person number`.

```
METADATA|User|PersonNumber|Username
MERGE|User|12312|carlton.baugh@vision.com
```

This example creates a request to suspend an active user.

```
METADATA|User|PersonNumber|Suspended
MERGE|User|12312|Y
```

This example creates a request to activate a suspended user.

```
METADATA|User|PersonNumber|Suspended
MERGE|User|12312|N
```

This example updates the `CredentialsEmailSent` indicator, which determines whether an email containing the user credentials is sent to the user. If the `CredentialsEmailSent` indicator is `Y`, then no email is sent. If the indicator is `N`, then an email is sent. In this example, the indicator is changed from `Y` to `N`.

```
METADATA|User|PersonNumber|CredentialsEmailSent
MERGE|User|12312|N
```

Updating User Roles

You use the User component to add a single role. Note that you supply role codes rather than role names. For example, you specify `ORA_PER_EMPLOYEE_ABSTRACT` rather than Employee to add the employee role. If you supply an invalid role code, then the request is created but any attempt to provision the role fails.

This example creates a request to add a single role to the user.

```
METADATA|User|PersonNumber|RoleCommonName|AddRemoveRole
MERGE|User|12312|ORA_PER_EMPLOYEE_ABSTRACT|ADD
```

This example creates a request to add multiple roles to the user. You use the User Role component to add multiple roles.

```
METADATA|User|PersonNumber
MERGE|User|12312
METADATA|UserRole|PersonNumber|RoleCommonName|AddRemoveRole
MERGE|UserRole|12312|ORA_PER_EMPLOYEE_ABSTRACT|ADD
MERGE|UserRole|12312|ORA_PER_LINE_MANAGER_ABSTRACT|ADD
```

This example creates a request to remove a single role from the user. You use the User component to remove a single role.

```
METADATA|User|PersonNumber|RoleCommonName|AddRemoveRole
MERGE|User|12312|ORA_PER_EMPLOYEE_ABSTRACT|REMOVE
```

This example creates a request to remove multiple roles from the user. You use the User Role component to remove multiple roles.

```
METADATA|User|PersonNumber
MERGE|User|12312
METADATA|UserRole|PersonNumber|RoleCommonName|AddRemoveRole
MERGE|UserRole|12312|ORA_PER_EMPLOYEE_ABSTRACT|REMOVE
MERGE|UserRole|12312|ORA_PER_LINE_MANAGER_ABSTRACT|REMOVE
```
Creating a User Account

You can use the User object to create a user account in limited circumstances. For example:

- An attempt to request a user account when loading the Worker object has failed.
- The `GeneratedUserAccountFlag` attribute of the Person User Information component was set to `N` either by mistake or intentionally for the loaded worker.

Creating a user account is a two-step process:

1. Create the user.
2. Activate the user and add roles.

This example creates a request to create a user account for the specified person number. You can also supply a user name if required. If you supply no user name, then the user name is generated in the default format as specified on the Security Console. Once the account exists, it’s suspended immediately as it has no roles.

```
METADATA|User|PersonNumber|GenerateUserAccount
MERGE|User|12312|Y
```

This example activates the user account and adds roles so that the account remains active.

```
METADATA|User|PersonNumber|Suspended
METADATA|UserRole|PersonNumber|RoleCommonName|AddRemoveRole
MERGE|User|12312|N
MERGE|UserRole|12312|ORA_PER_EMPLOYEE_ABSTRACT|ADD
MERGE|UserRole|12312|ORA_PER_LINE_MANAGER_ABSTRACT|ADD
```

Tip: User-update requests aren’t processed until Send Pending LDAP Requests next runs. If the process isn’t scheduled, remember to run it.

Deleting a User Account

This example deletes the user account for the worker with the specified person number.

```
METADATA|User|PersonNumber
DELETE|User|12312
```

Guidelines for Loading Worker Schedules

A worker schedule defines a worker’s shift pattern for a specified period. To load worker schedules using HCM Data Loader, use the Schedule Request object. This topic describes some aspects of worker schedules that you must understand to load them successfully.

Preparing to Load Worker Schedules

Before you load worker schedules, you must define shift codes and scheduler profiles. You can also define shift layouts, if required. This table identifies the relevant tasks. Perform these tasks in the Time Management work area.
## Chapter 10

### Loading Worker-Related Objects

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage Shift Properties</td>
<td>Defines shift codes for all shifts</td>
</tr>
<tr>
<td>Manage Scheduler Profiles</td>
<td>Defines scheduler profiles. Ensure that each of the workers whose shifts you load appears in a scheduler profile.</td>
</tr>
<tr>
<td>Manage Layout Sets</td>
<td>Defines a shift layout for loaded time attributes. This task is optional. If you don’t define the shift layout, then you can still load time attributes but you can’t see them in the application.</td>
</tr>
</tbody>
</table>

You can also set up and enable notifications for schedulers and workers. This table identifies the notifications. Configure these notifications in the Alerts Composer work area.

<table>
<thead>
<tr>
<th>Notification</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTS Worker Shifts Imported</td>
<td>Sent to managers and schedulers when a new schedule is loaded</td>
</tr>
<tr>
<td>HTS Schedule Publication</td>
<td>Sent to workers when a new schedule is published</td>
</tr>
</tbody>
</table>

**Tip:** You can also enable or disable worker notifications by setting the `WorkerNotification` attribute of the Schedule Event component when you load schedule requests.

### How Worker Schedules Are Loaded

Unlike most other HCM Data Loader business objects, worker schedules aren’t loaded from the HCM Data Loader stage tables to the application tables. Instead, worker schedules are loaded from the HCM Data Loader stage tables to the schedule stage tables. You must then run the `Process Imported Shifts` process to load worker schedules into either the planned or the planned and published schedule tables. You run the `Process Imported Shifts` process using the Manage Scheduled Processes task in the Time Management work area. This process:

- Merges all schedule events for a worker from HCM Data Loader and REST services
- Consolidates the schedule events for a worker from multiple schedule requests, orders them by request time, and processes them in this order

This approach to loading Schedule Request objects maintains data integrity. However, you see only basic validation errors from HCM Data Loader. Most validation is performed when `Process Imported Shifts` loads Schedule Request objects into the schedule tables.

### Import Modes

When loading Schedule Request objects, you must set the `ImportMode` attribute of the Schedule Event component to either **FULL** or **UPDATE**. This table describes the import modes.

<table>
<thead>
<tr>
<th>Import Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FULL</strong></td>
<td>Use <strong>FULL</strong> mode when you create worker schedules. For example, use this mode to load worker schedules during implementation. When the mode is <strong>FULL</strong>, you must set the <code>ShiftAction</code> attribute of all Schedule Shift Event components in the Schedule Event to <strong>CREATE</strong>.</td>
</tr>
</tbody>
</table>
Verifying Loaded Schedules

Time and labor managers and schedulers verify loaded schedules in the Time Management work area. They verify:

- Planned schedules on the Manage Planned Schedule page
- Published schedules using the View Published Schedule task

Workers view their published schedules using the calendar and Team Schedule pages of the Time work area.

Examples of Creating Worker Schedules

This topic provides examples showing how to create worker schedules using HCM Data Loader.

Creating a Schedule for a Worker for a Specific Week

For each worker whose schedule you’re creating, you must load Schedule Request, Schedule Event, and Schedule Shift Event components. On the Schedule Event component, you set the ImportMode attribute to FULL. On the Schedule Shift Event components, you set the ShiftAction attribute to CREATE.

This example ScheduleRequest.dat file creates a schedule for a worker for a specified week. It uses user keys.

```
METADATA|ScheduleRequest|ScheduleRequestNumber|RequestSource|RequestTime
MERGE|ScheduleRequest|SR1234501|3RD_PARTY_ABC|2017-01-01T13:25:20.010+01:00
METADATA|ScheduleEvent|ScheduleRequestNumber|ScheduleEventNumber|ImportMode|PersonNumber|PeriodStartDate|PeriodEndDate|AllowEdits|Publish|WorkerNotification
MERGE|ScheduleEvent|SR1234501|SE001|FULL|95160008182092|2017/01/01|2017/01/07|Y|Y|N
MERGE|ScheduleShiftEvent|ScheduleRequestNumber|ScheduleEventNumber|ScheduleShiftEventNumber|ShiftAction|ReferenceDay|ShiftStartTime|ShiftEndTime|ShiftTimeNotWorked|ShiftCode|ShiftCategory|ShiftType|AllowEdits
MERGE|ScheduleShiftEvent|SR1234501|SE001|SSE101|CREATE|2017/01/01|2017-01-01T09:00:00+01:00|2017-01-01T17:00:00+01:00|60|MORNTIME|WORK|TIME|N
MERGE|ScheduleShiftEvent|SR1234501|SE001|SSE102|CREATE|2017/01/02|2017-01-02T09:00:00+01:00|2017-01-02T17:00:00+01:00|60|MORNTIME|WORK|TIME|N
MERGE|ScheduleShiftEvent|SR1234501|SE001|SSE103|CREATE|2017/01/03|2017-01-03T09:00:00+01:00|2017-01-03T17:00:00+01:00|60|MORNTIME|WORK|TIME|N
MERGE|ScheduleShiftEvent|SR1234501|SE001|SSE104|CREATE|2017/01/04|2017-01-04T09:00:00+01:00|2017-01-04T17:00:00+01:00|60|MORNTIME|WORK|TIME|N
MERGE|ScheduleShiftEvent|SR1234501|SE001|SSE105|CREATE|2017/01/05|2017-01-05T09:00:00+01:00|2017-01-05T17:00:00+01:00|60|MORNTIME|WORK|TIME|N
METADATA|ScheduleShiftAttribute|ScheduleRequestNumber|ScheduleEventNumber|ScheduleShiftEventNumber|ScheduleShiftAttributeNumber|AttributeName|AttributeValue
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE101|SSA001|PayrollTimeType|WFM_PAY_REGULAR_US
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE101|SSA002|GD_Department_CHAR|1000
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE102|SSA001|PayrollTimeType|WFM_PAY_REGULAR_US
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE102|SSA002|GD_Department_CHAR|1000
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE103|SSA001|PayrollTimeType|WFM_PAY_REGULAR_US
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE103|SSA002|GD_Department_CHAR|1000
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE104|SSA001|PayrollTimeType|WFM_PAY_REGULAR_US
```
Creating a Shift for a Worker for a Specific Day

To create a single shift for a worker, you load the Schedule Request, Schedule Event, and Schedule Shift Event components. On the Schedule Event component, you set the **ImportMode** attribute to **FULL**. On the Schedule Shift Event component, you set the **ShiftAction** attribute to **CREATE**.

This example ScheduleRequest.dat file creates a shift for a worker on a specific day. It uses user keys.

```
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE104|SSA002|GD_Department_CHAR|1000
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE105|SSA001|PayrollTimeType|WFM_PAY_REGULAR_US
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE105|SSA002|GD_Department_CHAR|1000
```

### Note:
Shift numbers are optional when you create a schedule. Typically, they're generated by a third-party scheduling system. You must specify a shift number when updating or deleting a shift.

Examples of Updating Worker Schedules

This topic provides examples showing how to update existing worker schedules using HCM Data Loader.

### Updating a Shift

To update an existing shift, you load the Schedule Request, Schedule Event, and Schedule Shift Event components. On the Schedule Event component, you set the **ImportMode** attribute to **UPDATE**. On the Schedule Shift Event component, you specify the shift number and set the **ShiftAction** attribute to **UPDATE**.

This example ScheduleRequest.dat file updates an existing worker shift from a third-party application.

```
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE104|SSA002|GD_Department_CHAR|1000
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE105|SSA001|PayrollTimeType|WFM_PAY_REGULAR_US
MERGE|ScheduleShiftAttribute|SR1234501|SE001|SSE105|SSA002|GD_Department_CHAR|1000
```
Deleting a Shift

To delete a shift, you load the Schedule Request, Schedule Event, and Schedule Shift Event components. On the Schedule Event component, you set the **ImportMode** attribute to **UPDATE**. On the Schedule Shift Event component, you specify the shift number and set the **ShiftAction** attribute to **DELETE**.

This example ScheduleRequest.dat file deletes an existing shift

```
MERGE|ScheduleShiftEvent|SR1234504|SE001|SSE101|SN001|UPDATE|2017/01/01|2017-01-01T07:00:00+01:00|
2017-01-01T11:00:00+01:00|60|MORNTIME|WORK|TIME|N
```

Clearing a Schedule

To clear a worker schedule, load the Schedule Request and Schedule Event components. On the Schedule Event component, you set the **ImportMode** attribute to **FULL**. Because you aren’t providing Schedule Shift Event components, the Process Imported Shifts process deletes all shifts for the given period.

This example ScheduleRequest.dat file clears all existing shifts for a worker for a specified period.

```
MERGE|ScheduleShiftEvent|SR1234504|SE001|SSE101|SN001|DELETE|2017/01/01|2017-01-01T07:00:00+01:00|
2017-01-01T11:00:00+01:00||60|MORNTIME|WORK|TIME|N
```

FAQs for Loading Worker-Related Objects

Can I load allocated checklists using HCM Data Loader?

Yes. Both the checklist template and the worker to whom the checklist is allocated must exist in the target environment.
Can checklists be allocated automatically to workers created using HCM Data Loader?

No. To allocate a checklist to a worker, you can:

- Load an Allocated Checklist object for the worker.
- Allocate the checklist to the worker in the user interface.
Loading Work Structures

Overview of Loading Work Structures

The work structures that you can load using HCM Data Loader include grades, grade ladders, grade rates, jobs, job families, locations, organizations, positions, and HCM position hierarchies. This topic describes some general considerations that apply to most or all work structures.

Reviewing Component Hierarchies

Select a work structure on the View Business Objects page to review its component hierarchy.

Loading Set-Enabled Work Structures

These work structures are set enabled:

- Departments
- Grades
- Jobs
- Locations

You can create them in a specific reference data set, which restricts their use to an individual business unit. Alternatively, you can create them in the common set, where they're generally available. You must ensure that referenced sets exist and are associated with relevant business units, if appropriate.

Using Action Reasons

You can associate action reasons with these work structures:

- Grades
- Grade rates
- Grade ladders
- Jobs
- Locations
- Organizations
- Positions

If you plan to use action reasons, then you must create them before you load work structures. In the Setup and Maintenance work area, use the following:

- Functional Area: Workforce Structures
Task: Manage Action Reasons

Related Topics
- Sources of Business-Object Information

Examples of Loading Collective Agreements

A collective agreement is a written agreement between an employer and a union or bargaining unit. It defines terms and conditions of employment for the represented workers. The terms used to refer to collective agreements may vary by country. Generally, the agreement is for a specified term. This topic describes aspects of the Collective Agreement object that you must understand to load them successfully using HCM Data Loader.

Preparing to Load Collective Agreements

A collective agreement is country-specific. Therefore, LegislationCode is a required attribute of the Collective Agreement object.

You can associate any combination of a union, a bargaining unit, and a legal employer with the collective agreement. For example, you can associate a collective agreement with a union or with both a union and a legal employer. Any union, bargaining unit, or legal employer associated with collective agreements must exist in the target environment before you load those collective agreements.

Creating Collective Agreements

You supply collective agreements data in the CollectiveAgreements.dat file for processing by HCM Data Loader. This example CollectiveAgreements.dat file creates a collective agreement using source keys. The collective agreement is associated with a union, a bargaining unit, and a legal employer.

```
MERGE|CollectiveAgreements|VISION|VISCAGR01|SEIU UHW Medical Social Workers (California, Northern)|Service Employees International Union|2010/01/01|4712/12/31|A|SEIU_UHW_MSW_CA_N|Vision Corporation|US|Collective Agreement covering the medical social workers in northern California affiliated to the Service Employees International Union.|BU_SEIU_UHW_MSW_CA_N|Vision Corporation|John Gorman|Vision Corporation|Jane Reifer
```

This example CollectiveAgreements.dat file creates a collective agreement using user keys. The collective agreement is associated with a union, a bargaining unit, and a legal employer.

```
MERGE|CollectiveAgreements|SEIU UHW Medical Social Workers (California, Northern)|Service Employees International Union|2010/01/01|4712/12/31|A|SEIU_UHW_MSW_CA_N|Vision Corporation|US|Collective Agreement covering the medical social workers in northern California affiliated to the Service Employees International Union.|BU_SEIU_UHW_MSW_CA_N|Vision Corporation|John Gorman|Vision Corporation|Jane Reifer
```

You can't load attachments for collective agreements using HCM Data Loader.
Updating Collective Agreements

You can’t change either the first effective start date or last effective end date for an existing collective agreement record. Therefore, create your collective agreement records with an effective start date on or before the start date of any record that references them.

If the collective agreement is linked to an assignment, then you can’t edit the IdentificationCode, LegislationCode, BargainingUnitCode, LegalEmployerName, or LegalEntityId attributes.

Loading Translated Collective Agreement Names

This example CollectiveAgreementsTranslation.dat file translates the name of an existing collective agreement. It identifies the existing record using source keys.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>CollectiveAgreementsTranslation</th>
<th>SourceSystemOwner</th>
<th>SourceSystemId</th>
<th>EffectiveStartDate</th>
<th>EffectiveEndDate</th>
<th>Language</th>
<th>CollectiveAgreementName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>CollectiveAgreementsTranslation</td>
<td>VISION</td>
<td>VISCAGR01</td>
<td>2001/01/01</td>
<td>4712/12/31</td>
<td>FR</td>
<td>SEIU UHW Travailleurs sociaux medicaux (Californie, Nord)</td>
<td>Convention collective couvrant les travailleurs sociaux medicaux dans le nord de la Californie affiliée à l’Union internationale des employés du service.</td>
</tr>
</tbody>
</table>

Related Topics

• How You Manage Collective Agreements

Guidelines for Loading Grades

A grade identifies a worker’s rank or level of compensation. You can create grades either with or without grade steps. This topic describes aspects of the Grade object that you must understand to load grades successfully.

Grade Steps

If you’re using grade steps, then load one Grade Step component for each grade step.

Changing Logical Start Dates for Grades

You can change the first effective start date for an existing grade. However, the new effective start date must be before the existing effective start date. Create Grade objects with effective start dates on or before the start dates of other objects that refer to your grades.

Loading Translated Grade and Grade-Step Names

Supply grade and grade-step names in the language of the user who’s loading them. You supply a GradeTranslation.dat file to translate grade names into other enabled languages once the grades exist. You supply a GradeStepTranslation.dat file to translate grade-step names.

This example GradeTranslation.dat file translates the names of existing grades. It identifies the grades by their source keys.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>GradeTranslation</th>
<th>SourceSystemOwner</th>
<th>SourceSystemId</th>
<th>EffectiveStartDate</th>
<th>Language</th>
<th>GradeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>GradeTranslation</td>
<td>VISION</td>
<td>ADMIN</td>
<td>1951/01/01</td>
<td>FR</td>
<td>Administrateur</td>
</tr>
<tr>
<td>MERGE</td>
<td>GradeTranslation</td>
<td>VISION</td>
<td>IC1</td>
<td>1951/01/01</td>
<td>FR</td>
<td>Contributeur individuel</td>
</tr>
<tr>
<td>MERGE</td>
<td>GradeTranslation</td>
<td>VISION</td>
<td>M1</td>
<td>1951/01/01</td>
<td>FR</td>
<td>Superviseur</td>
</tr>
</tbody>
</table>
This example GradeStepTranslation.dat file translates the names of existing grade steps. It identifies the grade steps by their source keys.

METADATA|GradeStepTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|GradeStepName
METADATA|GradeStepTranslation|VISION|LT_CDR1|1951/01/01|FR|Niveau 1
METADATA|GradeStepTranslation|VISION|LT_CDR2|1951/01/01|FR|Niveau 2
METADATA|GradeStepTranslation|VISION|LT_CDR3|1951/01/01|FR|Niveau 3

Deleting Grades

You can delete a Grade object using HCM Data Loader, provided that it’s not referenced by other objects, such as Assignment, Job, or Grade Rate. This example Grade.dat file deletes existing grades and any associated grade steps. It identifies grades by their source keys.

METADATA|Grade|SourceSystemOwner|SourceSystemId|EffectiveStartDate
DELETE|Grade|VISION|CADET|1951/01/01
DELETE|Grade|VISION|LT_CDR|1951/01/01

Guidelines for Loading Grade Ladders

A grade ladder is a sequence of grades, where the grade order represents a worker’s progression. This topic describes aspects of the Grade Ladder object that you must understand to load grade ladders successfully.

The Grade-Ladder Component Hierarchy

The grade ladder is made up of these components:

- Grade Ladder.
- Grades in Ladder. Include one component of this type for each grade, regardless of the grade ladder type.
- Step Rate. Include at least one component of this type for grade ladders of type STEP.
- Step Rate Value. Include one component of this type for each step in a grade ladder of type STEP.

Loading Translated Grade-Ladder and Step-Rate Names

Supply grade-ladder and step-rate names in the language of the user who’s loading them. You supply a GradeLadderTranslation.dat file to translate grade-ladder names into other enabled languages once the grade ladders exist. You supply a StepRateTranslation.dat file to translate step-rate names. This example GradeLadderTranslation.dat file translates the name of an existing grade ladder. It identifies the grade ladder by its source key.

METADATA|GradeLadderTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|GradeLadderName
METADATA|GradeLadderTranslation|VISION|GRD_LDR_OFF|1951/01/01|FR|Officier

Deleting Grade Ladders

You can delete a Grade Ladder object using HCM Data Loader, provided that it’s not referenced by another object, such as Assignment. Deleting a grade ladder doesn’t delete the associated grades. However, if you delete a grade ladder with steps, then the associated step rates and step-rate values are deleted automatically. This example GradeLadder.dat file deletes an existing grade ladder. It identifies the grade ladder by its source key.
Guidelines for Loading Grade Rates

A grade rate is a collection of general information about a payment type, such as salary, bonus, or overtime. It includes details such as the currency, frequency of the payment, and annualization factor. For example, the Overtime Rate Analysts grade rate could provide this information for the overtime payments made to analysts. Grade rates are associated with grade-rate values. Each associated grade-rate value identifies a grade and either a fixed payment or a payment range for the grade. For a payment range, you must supply both the minimum and maximum values. This topic describes aspects of the Grade Rate object that you must understand to load grade rates successfully.

Loading Referenced Objects

Grade rates and grade-rate values are created for a legislative data group. Therefore, Legislative Data Group objects must exist before you load grade rates.

Loading Translated Grade-Rate Names

Supply grade-rate names in the language of the user who’s loading them. You supply a GradeRateTranslation.dat file to translate grade-rate names into other enabled languages once the grade rates exist. This example translates the name of an existing grade rate. It identifies the grade rate by its source key.

Deleting Grade Rates

You can delete a Grade Rate object using HCM Data Loader, provided that it’s not referenced by another object, such as Salary Basis. If you delete a grade rate, then its associated grade-rate values are deleted automatically. This example GradeRate.dat file deletes an existing grade rate and its grade-rate values. It identifies the grade rate by its source key.

Guidelines for Loading Jobs

A job defines the worker’s role in an organization in general terms. For example, a worker’s job could be payroll manager, sales consultant, or administrator. This topic describes aspects of the job object that you must understand to load Job objects successfully.

Loading Referenced Objects

If you’re associating either job families or valid grades with your jobs, then you must load those objects before you load the jobs.
Benchmark Jobs

To use benchmark jobs:

1. Load any benchmark jobs with the `BenchmarkJobFlag` attribute set to `Yes`. Benchmark jobs can’t refer to other benchmark jobs. Therefore, you can’t load benchmark jobs with either a `BenchmarkJobId` or `BenchmarkJobCode` value.

2. Load the remaining jobs. For those jobs that refer to a benchmark job, use the `BenchmarkJobId` or `BenchmarkJobCode` attribute to identify the relevant benchmark job. A job can’t be its own benchmark job. Therefore, these values must not identify the job that you’re loading.

Loading Translated Job Names

Supply job names in the language of the user who’s loading them. You supply a `JobTranslation.dat` file to translate job names into other enabled languages once the jobs exist. This example translates the names of existing jobs. It identifies those jobs by their source keys.

```
METADATA|JobTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|Name
MERGE|JobTranslation|VISION|SALES_CONS|2000/01/01|FR|Consultant en Ventes
MERGE|JobTranslation|VISION|SALES_MGR|2000/01/01|FR|Superviseur des Ventes
MERGE|JobTranslation|VISION|SALES_DIR|2000/01/01|FR|Directeur des Ventes
```

Deleting Jobs

You can’t delete Job objects using HCM Data Loader. However, you can make jobs inactive. This example `Job.dat` file inactivates an existing job. It identifies the job by its source key.

```
METADATA|Job|SourceSystemOwner|SourceSystemId|EffectiveStartDate|ActiveStatus
MERGE|Job|VISION|MRKT_CONS|2015/01/01|I
```

Examples of Loading Job Families

A job family groups similar or related jobs for ease of reporting. For example, you could add the Trust Analyst and Operations Analyst jobs to the Analyst job family. This topic provides some examples showing how to load and manage Job Family objects.

Creating Job Families Using Source Keys

This example `JobFamily.dat` file creates several job families that are identified using source keys.

```
METADATA|JobFamily|SourceSystemOwner|SourceSystemId|EffectiveStartDate|JobFamilyCode|JobFamilyName|ActiveStatus
MERGE|JobFamily|VISION|ANALYST|2000/01/01|ANALYST|Analyst|A
MERGE|JobFamily|VISION|TECHNICIAN|2000/01/01|TECHNICIAN|Technician|A
MERGE|JobFamily|VISION|CONSULT|2000/01/01|CONSULTANT|Consultant|A
MERGE|JobFamily|VISION|MANAGEMENT|2000/01/01|MANAGEMENT|Management|A
```
Loading Translated Job-Family Names

Job family names must be unique in the enterprise. Supply job family names in the language of the user who’s loading them. You supply a JobFamilyTranslation.dat file to translate job-family names into other enabled languages once the job families exist. This example translates the names of existing job families. The job families are identified by their source keys.

```
METADATA|JobFamilyTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|JobFamilyName
MERGE|JobFamilyTranslation|VISION|ANALYST|2000/01/01|FR|Analyste
MERGE|JobFamilyTranslation|VISION|TECHNICIAN|2000/01/01|FR|Technicien
```

Deleting Job Families

You can delete a Job Family object using HCM Data Loader, provided that the job family isn’t referenced by any job. This example deletes a job family. It identifies the job family by its source key.

```
METADATA|JobFamily|SourceSystemOwner|SourceSystemId|EffectiveStartDate
DELETE|JobFamily|VISION|TECHNICIAN|2000/01/01
```

Guidelines for Loading Locations

A location is the physical address of a place where you conduct business or that’s of interest to your business. The Location object records the address and other details, such as the location name, description, phones, and billing and shipping information. This topic describes aspects of the Location object that you must understand to load locations successfully.

Changing Logical Start and End Dates for Locations

You can’t change either the first effective start date or last effective end date for an existing location. Create location records with effective start dates on or before the start dates of other objects, such as Organization, that refer to your locations.

Loading Translated Location Names

Supply location names and descriptions in the language of the user who’s loading them. You supply a LocationTranslation.dat file to translate location names and descriptions into other enabled languages once the locations exist. This example translates the name and description of an existing location. It identifies the location by its source key.

```
METADATA|LocationTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|LocationName|Description
MERGE|LocationTranslation|VISION|LOC_RD_UK|1951/01/01|FR|Vision Corporation Recherche et Developpement|Centre de recherche et de developpement
```

Deleting Locations

You can’t delete Location objects using HCM Data Loader. However, you can make locations inactive. This example Location.dat file makes a location inactive as of 01/01/15. The location is identified by its source key.

```
METADATA|Location|SourceSystemOwner|SourceSystemId|EffectiveStartDate|ActiveStatus
MERGE|Location|VISION|LOC_uk_100|2015/01/01|I
```
Guidelines for Loading Organizations

An organization can be any organizational unit, such as a department or division, in your enterprise. Organizations provide the framework for performing legal reporting, financial control, and management reporting. This topic describes aspects of the Organization object that you must understand to load organizations successfully.

Organization Components

A single organization can have multiple classifications. For example, a division can also be a department. Therefore, you can load multiple Organization Classification components for a single Organization object.

Note: You must load any Organization Extra Information component with the parent Organization component in the same Organization.dat file. You can’t load Organization Extra Information components independently of the parent Organization component. You can load other child components either when creating the organization or once the organization exists.

Loading Referenced Objects

Before you load organizations, referenced objects must exist in the target environment. In particular:

- If you plan to associate locations with your organizations, then you must load the locations before you load the organizations.
- To associate a reporting manager with your organization, you must:
  a. Load the organization without identifying the reporting manager.
  b. Load the worker information for the reporting manager.
  c. Update your organization to add the reporting manager.

Tip: As a worker is hired into a legal employer, business unit, and department, you can’t load workers before you load organizations.

Note: Departments are set enabled. You must ensure that relevant reference data sets exist before you load departments. Other types of organizations aren’t set enabled.

Organization Classification Codes

An organization can have one or more classifications. You can supply either the code or its name. Valid classifications are shown in this table.

<table>
<thead>
<tr>
<th>Classification Name</th>
<th>Classification Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department</td>
<td>DEPARTMENT</td>
</tr>
<tr>
<td>Disability Organization</td>
<td>HCM_DISABILITY.Organization</td>
</tr>
</tbody>
</table>
Oracle Human Capital Management Cloud
Integrating with HCM

Chapter 11
Loading Work Structures

<table>
<thead>
<tr>
<th>Classification Name</th>
<th>Classification Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Division</td>
<td>HCM_DIVISION</td>
</tr>
<tr>
<td>Reporting Establishment</td>
<td>HCM_REPORTING_ESTABLISHMENT</td>
</tr>
<tr>
<td>Union</td>
<td>PER_WORKER_UNION</td>
</tr>
</tbody>
</table>

You load an Organization Classification component for each of the organization’s classifications, including the one identified on the Organization component.

Loading Translated Organization Names

Supply organization names in the language of the user who’s loading them. You supply an OrganizationTranslation.dat file to translate organization names into other enabled languages once the organizations exist. This example OrganizationTranslation.dat file translates the name of an existing organization. It identifies the organization by its source key.

```
METADATA|OrganizationTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|Name
MERGE|OrganizationTranslation|VISION|MRKT|2000/01/01|FR|Marketing
```

Deleting Organizations

You can’t delete Organization objects using HCM Data Loader, nor can you make them inactive. However, you can make an organization classification inactive, provided that it’s no longer used. For example, for an organization that’s classified as both a department and a division, you could:

- Make the division inactive but leave the department active, so that you can still continue to use the department.
- Deactivate the whole organization by making both classifications inactive.

This example Organization.dat file inactivates an existing organization classification. The organization is identified by its source key.

```
METADATA|Organization|SourceSystemOwner|SourceSystemId|EffectiveStartDate
MERGE|Organization|VISION|CLARITY|2015/01/01
METADATA|OrgUnitClassification|SourceSystemOwner|SourceSystemId|EffectiveStartDate|
OrganizationId(SourceSystemId)|Status
MERGE|OrgUnitClassification|VISION|CLARITY_DIS|2015/01/01|CLARITY_I
```

💡 Note: The file must include an Organization component with an effective start date equal to the date on which the classification becomes inactive.

Guidelines for Loading Positions

A position is a single occurrence of a job in a specified department. Positions may also be restricted by location. For example, you could create the HR Manager Sales France position for the HR Manager job in the Sales department in France. This topic describes aspects of the Position object that you must understand to load positions successfully.
Effective Start Dates
A Position object and its child components must have the same effective start date.

Loading Referenced Objects
Before you load positions, referenced objects must exist in the target environment. In particular:

- Positions exist in business units. Ensure that referenced business units exist.
- Both Job and Department are required attributes of a position. Ensure that referenced jobs and departments exist.
- You can associate a location and valid grades with a position. Ensure that any referenced locations and grades exist.

Position Type
When the position type is Single incumbent, you must supply:

- An FTE value. The value must be less than or equal to 1. For example, if the position incumbent works half the standard working hours, then the full-time equivalent value is 0.5.
- A headcount of 1.

When the position type is Shared, you must supply:

- An FTE value
- A headcount that's equal to or greater than 1

Loading Translated Position Names
Supply position names in the language of the user who’s loading them. You supply a PositionTranslation.dat file to translate position names into other enabled languages once the positions exist. This example PositionTranslation.dat file translates the name of an existing position. It identifies the position by its source key.

```
METADATA|PositionTranslation|SourceSystemOwner|SourceSystemId|EffectiveStartDate|Language|Name
MERGE|PositionTranslation|VISION|APP_SALES|2015/01/01|FR|Sales Consultant d’application
```

Deleting Positions
You can delete a Position object using HCM Data Loader, provided that it has never had an incumbent. When you delete a position, its child components are deleted automatically. This example Position.dat file deletes an existing position. The position is identified by its source key.

```
METADATA|Position|SourceSystemOwner|SourceSystemId|EffectiveStartDate
DELETE|Position|VISION|APP_SALES|2015/01/01
```

Examples of Loading HCM Position Hierarchies
An HCM position hierarchy is derived from the relationships between positions and their parent positions. You can identify a position’s parent position on the Manage Positions page if the Use HCM Position Hierarchy option is enabled for the enterprise. This topic provides examples of how to load Position Hierarchy objects using HCM Data Loader. Each Position Hierarchy object identifies a position and its parent. All positions must exist in the target environment.
This figure shows the position hierarchy that these examples load.

Loading HCM Position Hierarchies Using Source Keys

This example PositionHierarchy.dat file loads Position Hierarchy objects using source keys.

```
METADATA|PositionHierarchy|SourceSystemOwner|SourceSystemId|EffectiveStartDate|EffectiveEndDate|
ParentPositionId(SourceSystemId)|PositionId(SourceSystemId)
MERGE|PositionHierarchy|Vision|1|2015/01/01|4712/12/31|PM_MGR_ACCESS|PM_ACCESS
MERGE|PositionHierarchy|Vision|2|2015/01/01|4712/12/31|PM_MGR_ACCESS|SNR_PM_ACCESS
MERGE|PositionHierarchy|Vision|3|2015/01/01|4712/12/31|PM_MGR_SALES|PM_SALES
MERGE|PositionHierarchy|Vision|4|2015/01/01|4712/12/31|PM_MGR_SALES|SNR_PM_SALES
MERGE|PositionHierarchy|Vision|5|2015/01/01|4712/12/31|PM_DIR|PM_MGR_ACCESS
MERGE|PositionHierarchy|Vision|6|2015/01/01|4712/12/31|PM_DIR|PM_MGR_SALES
MERGE|PositionHierarchy|Vision|7|2015/01/01|4712/12/31|VP|PM_DIR
MERGE|PositionHierarchy|Vision|8|2015/01/01|4712/12/31|EVP|VP
```
Loading HCM Position Hierarchies Using User Keys

This example PositionHierarchy.dat file loads Position Hierarchy objects using user keys.

```
METADATA|PositionHierarchy|EffectiveStartDate|EffectiveEndDate|ParentBusinessUnitName|ParentPositionCode|BusinessUnitName|PositionCode
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|PM_MGR_ACCESS|VisionBU|PM_ACCESS
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|PM_MGR_ACCESS|VisionBU|SNR_PM_ACCESS
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|PM_MGR_SALES|VisionBU|PM_SALES
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|PM_MGR_SALES|VisionBU|SNR_PM_SALES
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|PM_DIR|VisionBU|PM_MGR_ACCESS
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|PM_DIR|VisionBU|PM_MGR_SALES
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|VP|VisionBU|PM_DIR
MERGE|PositionHierarchy|2015/01/01|4712/12/31|VisionBU|EVP|VisionBU|VP
```

Running the Flatten HCM Position Hierarchy Process

Changes to the parent position on the Manage Positions page automatically trigger a process to update the position hierarchy. This process is also triggered automatically when you load position hierarchies using HCM Data Loader. You can prevent the Flatten HCM Position Hierarchy process from running automatically by including a SET instruction in the PositionHierarchy.dat file.

Guidelines for Loading Department Trees

A department tree is a hierarchy of departments. For each tree, you can define multiple versions. However, only one version can be active on any date. This topic describes aspects of Department Tree objects that you must understand to load them successfully using HCM Data Loader.

Key Support

Department Tree is unlike most other objects that you can load using HCM Data Loader, as it doesn’t support the integration-key types. When you load department trees, you must supply the user-key attributes.

How Department Trees Are Implemented

Department trees are implemented using two HCM Data Loader objects. This table introduces those objects.

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department Tree</td>
<td>The definition of a tree and the versions of the tree. A tree version is empty until you load department tree nodes for the tree version.</td>
</tr>
<tr>
<td>Department Tree Node</td>
<td>The department tree nodes for a tree version. Each node represents a department.</td>
</tr>
</tbody>
</table>
Preparing Department Trees for Use

After loading department tree nodes, you must flatten, audit, and activate your tree version. You can’t reference your tree version from other objects, such as Area of Responsibility, until it’s active. You can activate your tree version on the Manage Department Trees page in the Workforce Structures work area. In summary, you:

1. Search for and select your department tree.
2. Expand the hierarchy and select the version to activate.
3. Select Actions > Set Status > Active to activate the tree version. The tree flattening and audit occur automatically.

For more information about managing hierarchy trees, see the Implementing Global Human Resources guide.

Loading Department Trees

You supply department-tree data in the DepartmentTree.dat file for processing by HCM Data Loader. The TreeStructureCode attribute of both the department tree and the department tree version must be set to PER_DEPT_TREE_STRUCTURE for a department tree.

Note: When you load a new department tree version for an existing department tree, you must include the Department Tree component in the DepartmentTree.dat file. You can’t load a new department tree version in isolation.

This example DepartmentTree.dat file creates a department tree with code Department Tree Code and three versions, one for each of January, February, and March.

METADATA|DepartmentTree|TreeStructureCode|TreeCode|TreeName|TreeDescription
MERGE|DepartmentTree|PER_DEPT_TREE_STRUCTURE|Department Tree Code|My Tree Name|My Tree Description
METADATA|DepartmentTreeVersion|TreeStructureCode|TreeCode|TreeVersionName|TreeVersionDescription|TreeVersionStartDate|TreeVersionEndDate
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|Tree Description|Jan|2015/01/01|2015/01/31
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Feb|Tree Description|Feb|2015/02/01|2015/02/28
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Mar|Tree Description|Mar|2015/03/01|2015/03/31

Deleting Department Trees

You can delete Department Tree and Department Tree Version components using HCM Data Loader. When you delete a department tree, the tree and all of its versions and nodes are deleted.

Note: The departments themselves aren’t deleted. Only the relationships defined in the tree structure are deleted.

This example DepartmentTree.dat file deletes a department tree:

METADATA|DepartmentTree|TreeStructureCode|TreeCode
DELETE|DepartmentTree|PER_DEPT_TREE_STRUCTURE|Department Tree Code

If you delete a department tree version, then only the specified version and its nodes are deleted. The tree and its other versions are retained. This example DepartmentTree.dat file deletes a department tree version:

METADATA|DepartmentTreeVersion|TreeStructureCode|TreeCode|TreeVersionName
DELETE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Feb
Guidelines for Loading Department Tree Nodes

A department is an internal organization to which workers are assigned. You can add any organization that has the Department classification as a tree node to a department tree version. This topic describes aspects of Department Tree Node objects that you must understand to load them successfully using HCM Data Loader.

Preparing to Load Department Tree Nodes

Before you load department tree nodes:

- You must create the department tree and department tree version to which the nodes belong. You can either load them using HCM Data Loader or create them using the Manage Department Trees task in the Workforce Structures work area.
- The departments that the tree nodes refer to must exist on the start date of the tree hierarchy version. Simply adding a department to the department tree hierarchy doesn’t create the department. Similarly, removing a department reference from a department tree hierarchy doesn’t delete the department record.

Loading Department Tree Nodes

The department tree node records the relationship between a parent department and a child, which can be either a department or another department tree version. When the child is another department tree version, the department nodes of that version are included by reference. When you load department tree nodes, these rules apply:

- You must supply a parent department for all nodes except the top-level node of your department tree version. The parent departments must exist.
- The TreeStructureCode attribute must be set to PER_DEPT_TREE_STRUCTURE.
- To avoid circular relationships, a department can appear only once in each tree version.
- Each tree version can have only one top node. If you need multiple peer departments at the top, then consider using a dummy department as the top node.

Related Topics

- Processes to Run After Loading Data

Examples of Loading Department Tree Nodes

You can load any organization that has the Department classification as a tree node to a department tree version. This topic shows how to load Department Tree Node objects using HCM Data Loader.

Creating Department Tree Nodes

This example DepartmentTreeNode.dat file loads department tree nodes to an existing department tree version.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>DepartmentTreeNode</th>
<th>TreeStructureCode</th>
<th>TreeCode</th>
<th>TreeVersionName</th>
<th>DepartmentName</th>
<th>ParentDepartmentName</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>DepartmentTreeVersion</td>
<td>PER_DEPT_TREE_STRUCTURE</td>
<td>Department Tree Code</td>
<td>Tree Version Jan</td>
<td>Vision University</td>
<td></td>
</tr>
</tbody>
</table>
This figure shows the hierarchy nodes that this DepartmentTreeNode.dat file creates.

Deleting Department Tree Nodes

You can delete Department Tree Node objects using HCM Data Loader. The departments are removed from the tree version, but the department objects themselves continue to exist. To delete a department tree node and all nodes below it in the department tree hierarchy, set the `DeleteChildNodesFlag` attribute to `Y`. This example DepartmentTreeNode.dat file deletes a node and its child nodes.

```
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|School of Arts and Science|Vision University
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|Drama|School of Arts and Science
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|Languages|Vision University
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|French|Languages
MERGE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|German|Languages

METADATA|DepartmentTreeNode|TreeStructureCode|TreeCode|TreeVersionName|DepartmentName|DeleteChildNodesFlag
DELETE|DepartmentTreeVersion|PER_DEPT_TREE_STRUCTURE|Department Tree Code|Tree Version Jan|Languages|Y
```
This figure shows the hierarchy after deleting the Languages node and its child nodes.

To delete a department tree node and promote its child nodes, set the `DeleteChildNodesFlag` attribute to `N`. This example DepartmentTreeNode.dat file deletes a node and promotes its child nodes.

This figure shows the hierarchy after deleting the Languages node and promoting its child nodes.
Guidelines for Loading Organization Trees

An organization tree is a hierarchy of organizations of any classification. For each tree, you can define multiple versions. However, only one version can be active on any date. This topic describes aspects of Organization Tree objects that you must understand to load them successfully using HCM Data Loader.

Key Support

Organization Tree is unlike most other objects that you can load using HCM Data Loader, as it doesn’t support the integration-key types. When you load organization trees, you must supply the user-key attributes.

How Organization Trees Are Implemented

Organization trees are implemented using two HCM Data Loader objects. This table introduces those objects.

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization Tree</td>
<td>The definition of a tree and the versions of the tree. A tree version is empty until you load organization tree nodes for the tree version.</td>
</tr>
<tr>
<td>Organization Tree Node</td>
<td>The organization tree nodes for a tree version. Each node represents an organization.</td>
</tr>
</tbody>
</table>
Preparing Organization Trees for Use

After loading organization tree nodes, you must flatten, audit, and activate your tree version. You can’t reference your tree version from other objects, such as Area of Responsibility, until it’s active. You can activate your tree version on the Manage Organization Trees page in the Workforce Structures work area. In summary, you:

1. Search for and select your organization tree.
2. Expand the hierarchy and select the version to activate.
3. Select Actions > Set Status > Active to activate the tree version. The tree flattening and audit occur automatically.

For more information about managing hierarchy trees, see the Implementing Global Human Resources guide.

Loading Organization Trees

You supply organization-tree data in the OrganizationTree.dat file for processing by HCM Data Loader. The TreeStructureCode attribute of both the organization tree and the organization tree version must be set to PER_ORG_TREE_STRUCTURE for an organization tree.

> **Note:** When you load a new organization tree version for an existing organization tree, you must include the organization tree component in the OrganizationTree.dat file. You can't load a new organization tree version in isolation.

This example OrganizationTree.dat file creates an organization tree with code Organization Tree Code and three versions, one for each of January, February, and March.

```
METADATA|OrganizationTree|TreeStructureCode|TreeCode|TreeName|TreeDescription
MERGE|OrganizationTree|PER_ORG_TREE_STRUCTURE|Organization Tree Code|My Tree Name|My Tree Description
METADATA|OrganizationTreeVersion|TreeStructureCode|TreeCode|TreeVersionName|TreeVersionDescription|TreeVersionStartDate|TreeVersionEndDate
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Organization Tree Code|Tree Version Jan|Tree Description Jan|2015/01/01|2015/01/31
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Organization Tree Code|Tree Version Feb|Tree Description Feb|2015/02/01|2015/02/28
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Organization Tree Code|Tree Version Mar|Tree Description Mar|2015/03/01|2015/03/31
```

Deleting Organization Trees

You can delete Organization Tree and Organization Tree Version components using HCM Data Loader. When you delete an organization tree, the tree and all of its versions and nodes are deleted.

> **Note:** The organizations themselves aren't deleted. Only the relationships defined in the tree structure are deleted.

This example OrganizationTree.dat file deletes an organization tree:

```
METADATA|OrganizationTree|TreeStructureCode|TreeCode
DELETE|OrganizationTree|PER_ORG_TREE_STRUCTURE|Organization Tree Code
```

If you delete an organization tree version, then only the specified version and its nodes are deleted. The tree and its other versions are retained. This example OrganizationTree.dat file deletes an organization tree version:

```
METADATA|OrganizationTreeVersion|TreeStructureCode|TreeCode|TreeVersionName
DELETE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Organization Tree Code|Tree Version Feb
```
Guidelines for Loading Organization Tree Nodes

You can load an organization of any classification as a tree node to an organization tree version. This topic describes aspects of Organization Tree Node objects that you must understand to load them successfully using HCM Data Loader.

Preparing to Load Organization Tree Nodes

Before you load organization tree nodes:

- You must create the organization tree and organization tree version to which the nodes belong. You can either load them using HCM Data Loader or create them using the Manage Organization Trees task in the Workforce Structures work area.
- The organizations that the tree nodes refer to must exist on the start date of the tree hierarchy version. Simply adding an organization to the organization tree hierarchy doesn't create the organization. Similarly, removing an organization reference from an organization tree hierarchy doesn't delete the organization record.

Loading Organization Tree Nodes

The organization tree node records the relationship between a parent organization and a child, which can be either an organization or another organization tree version. When the child is an organization, you specify its classification on the ClassificationCode attribute. When the child is another organization tree version, the organization nodes of that version are included by reference. When you load organization tree nodes, these rules apply:

- You must supply a parent organization for all nodes except the top-level node of your organization tree version. The parent organizations must exist.
- The TreeStructureCode attribute must be set to PER_ORG_TREE_STRUCTURE.
- To avoid circular relationships, an organization can appear only once in each tree version.
- Each tree version can have only one top node. If you need multiple peer organizations at the top, then consider using the enterprise as the top node.

Related Topics

- Processes to Run After Loading Data

Examples of Loading Organization Tree Nodes

You can load an organization with any classification as a tree node to an organization tree version. This topic shows how to load organization tree nodes using HCM Data Loader.

Creating Organization Tree Nodes

This example OrganizationTreeNode.dat file loads organization tree nodes to an existing organization tree version.

```
METADATA|OrganizationTreeNode|TreeStructureCode|TreeCode|TreeVersionName|OrganizationName|ClassificationCode|
ParentOrganizationName|ParentOrganizationClassificationCode
```
Deleting Organization Tree Nodes

You can delete organization tree nodes using HCM Data Loader. The organizations are removed from the tree version, but the organization objects themselves continue to exist. To delete an organization tree node and all nodes below it in the organization tree hierarchy, set the `DeleteChildNodesFlag` attribute to `Y`. This example OrganizationTreeNode.dat file deletes a node and its child nodes.

```plaintext
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Vision Tree Code|Tree Version 2017|Vision Enterprise|Enterprise|
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Vision Tree Code|Tree Version 2017|US Division|Division|Vision Enterprise|Enterprise
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Vision Tree Code|Tree Version 2017|EMEA Division|Division|Vision Enterprise|Enterprise
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Vision Tree Code|Tree Version 2017|France LE|Legal Employer|EMEA Division|Division
MERGE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Vision Tree Code|Tree Version 2017|Italy LE|Legal Employer|EMEA Division|Division
```

This figure shows the hierarchy nodes that this OrganizationTreeNode.dat file creates.
This figure shows the hierarchy after deleting the EMEA Division node and its child nodes.

To delete an organization tree node and promote its child nodes, set the `DeleteChildNodesFlag` attribute to `N`. This example `OrganizationTreeNode.dat` file deletes a node and promotes its child nodes.

```
METADATA|OrganizationTreeNode|TreeStructureCode|TreeCode|TreeVersionName|OrganizationName|ClassificationCode|
DeleteChildNodesFlag
DELETE|OrganizationTreeVersion|PER_ORG_TREE_STRUCTURE|Vision Tree Code|Tree Version 2017|EMEA Division|
Division|N
```

This figure shows the hierarchy after deleting the EMEA Division node and promoting its child nodes.
Guidelines for Loading Market Data Objects

You can load survey data from external suppliers using HCM Data Loader market data objects. The survey data can include salary, bonuses, benefits, and so on. This topic provides some tips for loading market data objects.

Loading Any Type of Market Data

Before you can load any of the market data objects, you must define the survey data supplier. Let’s say you’re loading survey data from Hay. You need a supplier definition for Hay in your environment before you can load the data. Use the Manage Supplier Structures task in the Compensation work area to define suppliers of survey data.

Loading Market Data Survey Data

Before you can load a Market Data Survey Data object, you must define:

- The survey, for example, Hay Professional. Use the Manage Surveys task in the Compensation work area to define surveys.
- The value of any CompensationTypeCode attribute that you supply. Use the Manage Compensation Types task in the Compensation work area to define compensation types.
- Any of these market data job structures objects that you reference on the Market Data Survey Data object:
  - Market Data Career Stream
  - Market Data Career Level
  - Market Data Other Level
  - Market Data Job Family
  - Market Data Job Function

You load just the market data job structures objects that you want to use in your compensation analyses.

- Any of these objects that you reference on the Market Data Survey Data object:
  - Market Data Job List
  - Market Data Location List

You need to load any job structures objects before the Market Data Job List object, because the job list refers to the job structures objects.

Tip: To be sure that all object references can resolve, load the Market Data Survey Data object last.

The objects referenced on the Market Data Survey Data object, such job family, can apply to more than one survey from the same supplier.
Loading Market Data Survey Data from Spreadsheets

You can use HCM Spreadsheet Data Loader to load market data objects. Spreadsheet templates are supplied, and you can copy them to create your own versions. You can find the supplied templates:

- In the Compensation work area, for Compensation Analysts
- In the Data Exchange work area, for Human Capital Management Integration Specialists

Examples of Loading Market Data Career Level, Career Stream, and Other Level

This topic shows how to load the Market Data Career Level, Market Data Career Stream, and Market Data Other Level objects using HCM Data Loader. A supplier of market data, such as Hay, supplies the object data.

Loading Market Data Career Level

A Market Data Career Level object describes a stage in a career, such as entry level, intermediate, professional, or executive. This example loads three Market Data Career Level objects. The objects are referenced using source keys.

```
METADATA|MktCareerLevel|SourceSystemOwner|SourceSystemId|VendorCode|CareerLevelCode|CareerLevelName|Description|CareerLevelSequence|Status|CareerLevelDescr
MERGE|MktCareerLevel|VISION|CL_HAY_APPR|HAY|APPR|Apprentice|Apprentice|1|A|HAY Apprentice Career Level Notes.txt
MERGE|MktCareerLevel|VISION|CL_HAY_JRNY|HAY|JRNY|Journeyman|Journeyman|2|A|HAY Journeyman Career Level Notes.txt
MERGE|MktCareerLevel|VISION|CL_HAY_MSTR|HAY|MSTR|Master|Master|3|A|HAY Master Career Level Notes.txt
```

You include the .txt files named on the CareerLevelDescr attribute in the ClobFiles folder in the .zip file.

Loading Market Data Career Stream

A Market Data Career Stream object is a model of a career, usually defined for a professional field. This example loads three Market Data Career Stream objects. The objects are referenced using source keys.

```
METADATA|MktCareerStream|SourceSystemOwner|SourceSystemId|VendorCode|CareerStreamCode|CareerStreamName|Description|CareerStreamSequence|Status|CareerStreamDescr
MERGE|MktCareerStream|VISION|CS_HAY_CONT|HAY|CONT|Contributing|Contributing|1|A|HAY Contributing Career Stream Notes.txt
MERGE|MktCareerStream|VISION|CS_HAY_JOUR|HAY|JOUR|Journey|Journey|2|A|HAY Journey Career Stream Notes.txt
MERGE|MktCareerStream|VISION|CS_HAY_ADV|HAY|ADVANCED|Advanced|Advanced|3|A|HAY Advanced Career Stream Notes.txt
```

You include the .txt files named on the CareerStreamDescr attribute in the ClobFiles folder in the .zip file.
Loading Market Data Other Level

A Market Data Other Level object can be any career stage that isn’t defined as a career level or career stream. This example loads four Market Data Other Level objects. The objects are referenced using source keys.

```
METADATA|MktOtherLevel|SourceSystemOwner|SourceSystemId|VendorCode|OtherLevelCode|OtherLevelName|Description|OtherLevelSequence|Status|OtherLevelDescr
MERGE|MktOtherLevel|VISION|OL_HAY_1|HAY|100|Associate|Associate|1|A|HAY Associate Other Level Notes.txt
MERGE|MktOtherLevel|VISION|OL_HAY_2|HAY|200|Analyst|Analyst|2|A|HAY Analyst Other Level Notes.txt
MERGE|MktOtherLevel|VISION|OL_HAY_3|HAY|300|Senior|Senior|3|A|HAY Senior Other Level Notes.txt
MERGE|MktOtherLevel|VISION|OL_HAY_4|HAY|400|Principal|Principal|4|A|HAY Principal Other Level Notes.txt
```

You include the .txt files named on the `OtherLevelDescr` attribute in the ClobFiles folder in the .zip file.

Examples of Loading Market Data Job Family and Function

This topic shows how to load the Market Data Job Family and Market Data Job Function objects using HCM Data Loader. A supplier of market survey data, such as Hay, supplies the object data.

Loading Market Data Job Family

You group similar jobs in Market Data Job Family objects to make reporting easier. For example, you could group the Trust Analyst and Operations Analyst jobs into a job family named Analyst. This example creates six Market Data Job Family objects. The objects are identified using source keys.

```
METADATA|MktJobFamily|SourceSystemOwner|SourceSystemId|VendorCode|JobFamilyCode|JobFamilyName|Description|JobFamilySequence|Status|JobFamilyDescr
MERGE|MktJobFamily|VISION|JF_HAY_HR-BEN|HAY|HR-BEN|Benefits|Human Resources - Benefits|1|A|HAY Benefits Job Family Notes.txt
MERGE|MktJobFamily|VISION|JF_HAY_HR-CMP|HAY|HR-CMP|Compensation|Human Resources - Compensation|2|A|HAY Compensation Job Family Notes.txt
MERGE|MktJobFamily|VISION|JF_HAY_HR-DEV|HAY|HR-DEV|Organizational Development|Human Resources - OD|3|A|HAY OD Job Family Notes.txt
MERGE|MktJobFamily|VISION|JF_HAY_AC-REC|HAY|AC-PAY|Accounts Receivable|Accounting - Accounts Receivable|4|A|HAY AR Job Family Notes.txt
MERGE|MktJobFamily|VISION|JF_HAY_AC-PAY|HAY|AC-PAY|Accounts Payable|Accounting - Accounts Payable|5|A|HAY AP Job Family Notes.txt
MERGE|MktJobFamily|VISION|JF_HAY_AC-COS|HAY|AC-COS|Cost Accounting|Accounting - Cost Accounting|6|A|HAY Cost Accounting Job Family Notes.txt
```

You include the .txt files named on the `JobFamilyDescr` attribute in the ClobFiles folder in the .zip file.

Loading Market Data Job Function

Market Data Job Function objects are broad categories for jobs, such as Accounting and Engineering. This example creates three Market Data Job Function objects. The objects are identified using source keys.

```
METADATA|MktJobFunction|SourceSystemOwner|SourceSystemId|VendorCode|JobFunctionCode|JobFunctionName|Description|JobFunctionSequence|Status|JobFunctionDescr
```
Examples of Loading Market Data Location List and Job List

This topic shows how to load the Market Data Location List and Market Data Job List objects using HCM Data Loader. A supplier of market data, such as Hay, provides the object data.

Loading Market Data Location List

A Market Data Location List object is a single location that’s included in the survey data. Its child component, Market Data Location Map, maps the survey location to an internal location. This example creates five locations with location mappings. The objects are identified using source keys.

```
MERGE|MktLocationList|VISION|LOC_HAY_AKR|HAY|AKR|Akron, OH|A
MERGE|MktLocationList|VISION|LOC_HAY_ALB|HAY|ALB|Albany, NY|A
MERGE|MktLocationList|VISION|LOC_HAY_ALL|HAY|ALL|Allentown, PA|A
MERGE|MktLocationList|VISION|LOC_HAY_ALQ|HAY|ALQ|Albuquerque, NM|A
MERGE|MktLocationList|VISION|LOC_HAY_CHI|HAY|CHI|Chicago, IL|A
```

The mappings between internal and external locations aren’t required if they’re one-to-one. But, they’re needed if you’re mapping multiple internal locations to an external location. You can create the mappings later in the user interface if you prefer.

Loading Market Data Job List

A Market Data Job List is information about a single job included the survey data. This example creates seven jobs. The objects are identified using source keys.

```
MERGE|MktJobList|VISION|HAY-ACCT_CA|HAY|ACCT_CA|Cost accountant match job notes|Cost Accountant|Cost Accountant job|A|VIS_PAY_CA|JF_HAY_AC-COS|JFN_HAY_ACCT||
```
Example of Loading Market Data Survey Data

The Market Data Survey Data object holds raw survey data. This topic shows how to load the Market Data Survey Data object using HCM Data Loader. A supplier of market survey data, such as Hay, provides the object data.

Guidelines for Loading Progression Grade Ladders

A progression grade ladder is a hierarchy of grades in sequence. It can include either grades with steps or grades without steps, but not both. If your grades have steps, then you define the grade rate using the Progression Grade Ladder object. If your grades don’t have steps, then you define the grade rate using the Progression Grade Rate object after loading your progression grade ladder records. This topic describes aspects of progression grade ladders that you must understand to load them successfully using HCM Data Loader.

Preparing to Load Progression Grade Ladders

These referenced objects must exist in the target environment:

- Grade and Grade Set
- Legislative Data Group

In addition, if you're using action reasons, then define or load them before loading progression grade ladders.
Using Grades Without Steps

If the grades don’t have steps, then set these attributes of the Progression Grade Ladder object as shown in this table:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GradeType</td>
<td>GRADE</td>
</tr>
<tr>
<td>ProgressionStyleCode</td>
<td>CMP_GSP_GP</td>
</tr>
</tbody>
</table>

Set both **ProgActionCode** and **SalaryActionCode**, which are required attributes, to the predefined value **CMP_GRADE_STEP_PROGRESSION**.

You load the separate Progression Grade Rate object instead of the Progression Step Rate and Progression Step Rate Value components of the Progression Grade Ladder object.

Using Grades with Steps

If the grades have steps, then set these attributes of the Progression Grade Ladder object as shown in this table:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GradeType</td>
<td>STEP</td>
</tr>
<tr>
<td>ProgressionStyleCode</td>
<td>CMP_GSP_GSP</td>
</tr>
</tbody>
</table>

Set both **ProgActionCode** and **SalaryActionCode**, which are required attributes, to the predefined value **CMP_GRADE_STEP_PROGRESSION**.

You load the Progression Step Rate and Progression Step Rate Value components of the Progression Grade Ladder object. You don't load the Progression Grade Rate object.

⚠️ Tip: Give the Progression Grade Ladder and Progression Step Rate components the same name, as a one-to-one relationship exists between them.

Translating Attribute Values

You can translate:

- The name of the progression grade ladder. Use the Progression Grade Ladder Translation object.
- The name of the progression step rate, if you’re using grades with steps. Use the Progression Step Rate Translation object.
- The name of the progression grade rate, if you’re using grades without steps. Use the Progression Grade Rate Translation object.

If you’re using the same name for the progression grade ladder and progression step rate, as recommended, then use the same translated value for both.
Using Fast Formulas

Set these attributes to CMP_GSP_DATE_RULE when their value is determined by fast formula:

- ProgressionDateCode
- SalaryChangeDateCode
- RateChangeDateCode

Specify the formula name on the ProgressionDateRuleName, SalaryChangeDateRuleName, or RateChangeDateRuleName attribute, as appropriate.

Set the SalaryCalculationMethodCode attribute to CMP_GSP_RATE_RULE when its value is determined by fast formula. Specify the formula name on the SalaryCalculationRuleName attribute.

The fast formulas must exist in the target environment before you load Progression Grade Ladder objects. Date formulas must have the Salary Progression Date Determination Rule type. Salary calculation formulas must have the Salary Progression Rate Calculation Rule type.

Deleting Progression Grade Ladders

You can delete a progression grade ladder, provided that it isn’t referenced by another object, for example, an assignment.

When you delete a progression grade ladder without steps, you must delete the associated Progression Grade Rate and Progression Rule objects separately. When you delete a progression grade ladder with steps, the associated step rates and values are deleted automatically. However, you must delete any associated Progression Rule object separately. When you delete any progression grade ladder, the associated grades and eligibility profiles are unaffected.

Examples of Loading Progression Grade Ladders with Steps

This topic provides examples showing how to load Progression Grade Ladder objects with steps.

Creating a Progression Grade Ladder with Steps Using Source Keys

This example ProgressionGradeLadder.dat file creates a progression grade ladder with steps. The file includes Progression Grade Ladder, Grades In Progression Ladder, Progression Step Rate, and Progression Step Rate Value components. All components use source keys. As recommended, the progression grade ladder and progression step rate have the same name.

This excerpt shows the Progression Grade Ladder component.
Creating a Progression Grade Ladder with Steps Using Fast Formulas and User Keys

This example ProgressionGradeLadder.dat file creates a progression grade ladder with steps. The file includes Progression Grade Ladder, Grades In Progression Ladder, Progression Step Rate, and Progression Step Rate Value components. All components use user keys. As recommended, the progression grade ladder and progression step rate have the same name.
This excerpt shows the Progression Grade Ladder component. It includes references to four existing fast formulas on the ProgressionDateCode, RateChangeDateCode, SalaryCalculationMethodCode, and SalaryChangeDateCode attributes.

This excerpt shows the Grades In Progression Ladder components.

This excerpt shows the Progression Step Rate component. The RateType value is SALARY and the ProgressionRateFlag attribute is set to Y to indicate that this rate is to be used for grade step progression.

This excerpt shows the Progression Step Rate Value components.

Deleting Progression Grade Ladders with Steps

This example ProgressionGradeLadder.dat file deletes the GSP Nurses progression grade ladder. It references the object by its user key. This file deletes the progression grade ladder, including the step rates and values. You must delete the associated Progression Rule object separately.
Examples of Loading Progression Grade Ladders Without Steps

This topic provides examples showing how to load Progression Grade Ladder objects without steps.

Creating a Progression Grade Ladder Without Steps Using Source Keys

This example ProgressionGradeLadder.dat file creates a progression grade ladder where the grades don’t have steps. The file includes Progression Grade Ladder and Grades In Progression Grade Ladder components. Both components use source keys. The rates are defined separately using Progression Rate objects.

This excerpt shows the Progression Grade Ladder component.

```
METADATA|ProgressionGradeLadder|SourceSystemOwner|SourceSystemId|EffectiveStartDate|GradeLadderName|GradeType|
ActiveStatus|GradeSetCode|AllowProgOverrideFlag|AllowSalaryOverrideFlag|AutoProgressionCode|AutoSalChangeCode|
ProgressionDateCode|ProgressionStyleCode|RateChangeDateCode|SalaryCalculationMethodCode|SalaryChangeDateCode|
UpdateSalaryFlag|ProgActionCode|SalaryActionCode|LegislativeDataGroupId(SourceSystemId)
MERGE|ProgressionGradeLadder|VISION|GSP_DEVELOPMENT|2000/02/15|GSP Development|GRADE|A|
COMMON|N|N|CMP_GSP_PROG_MANUAL|CMP_GSP_PROG_MANUAL|CMP_GSP_PROCESS_DT|CMP_GSP_GP|CMP_GSP_PROCESS_DT|
CMP_GSP_GRADE_STEP_RATE|CMP_GSP_ELIGIBILITY_DT|Y|CMP_GRADE_STEP_PROGRESSION|CMP_GRADE_STEP_PROGRESSION|
GSP_US_LDG1
```

This excerpt shows the Grades In Progression Grade Ladder components.

```
METADATA|GradesInProgressionLadder|SourceSystemOwner|SourceSystemId|EffectiveStartDate|GradesInLadderSequence|GradeSetCode|GradeId(SourceSystemId)|GradeLadderId(SourceSystemId)
MERGE|GradesInProgressionLadder|VISION|GSP_DEVELOPMENT_GIL1|2000/02/15|1|COMMON|DEV1_GSP|GSP_DEVELOPMENT
MERGE|GradesInProgressionLadder|VISION|GSP_DEVELOPMENT_GIL2|2000/02/15|2|COMMON|DEV2_GSP|GSP_DEVELOPMENT
MERGE|GradesInProgressionLadder|VISION|GSP_DEVELOPMENT_GIL3|2000/02/15|3|COMMON|DEV3_GSP|GSP_DEVELOPMENT
MERGE|GradesInProgressionLadder|VISION|GSP_DEVELOPMENT_GIL4|2000/02/15|4|COMMON|DEV4_GSP|GSP_DEVELOPMENT
MERGE|GradesInProgressionLadder|VISION|GSP_DEVELOPMENT_GIL5|2000/02/15|5|COMMON|DEV5_GSP|GSP_DEVELOPMENT
MERGE|GradesInProgressionLadder|VISION|GSP_DEVELOPMENT_GIL6|2000/02/15|6|COMMON|DEV6_GSP|GSP_DEVELOPMENT
```

Creating a Progression Grade Ladder Without Steps Using User Keys

This example ProgressionGradeLadder.dat file creates a progression grade ladder where the grades don’t have steps. The file includes Progression Grade Ladder and Grades In Progression Grade Ladder components. Both components use user keys. The rates are defined separately using Progression Rate objects.

This excerpt shows the Progression Grade Ladder component.

```
METADATA|ProgressionGradeLadder|EffectiveStartDate|GradeLadderName|GradeType|ActiveStatus|GradeSetCode|
AllowProgOverrideFlag|AllowSalaryOverrideFlag|AutoProgressionCode|AutoSalChangeCode|ProgressionDateCode|ProgressionStyleCode|RateChangeDateCode|SalaryCalculationMethodCode|SalaryChangeDateCode|
UpdateSalaryFlag|ProgActionCode|SalaryActionCode|LegislativeDataGroupId
MERGE|ProgressionGradeLadder|2010/01/01|GSP Staff|GRADE|A|COMMON|N|N|CMP_GSP_PROG_MANUAL|CMP_GSP_PROG_MANUAL|
CMP_GSP_PROCESS_DT|CMP_GSP_GP|CMP_GSP_PROCESS_DT|CMP_GSP_GRADE_STEP_RATE|CMP_GSP_ELIGIBILITY_DT|Y|
CMP_GRADE_STEP_PROGRESSION|CMP_GRADE_STEP_PROGRESSION|GBI United States LDG
```
This excerpt shows the Grades in Progression Grade Ladder components.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>GradesInProgressionLadder</th>
<th>EffectiveStartDate</th>
<th>GradesInLadderSequence</th>
<th>GradeSetCode</th>
<th>GradeCode</th>
<th>GradeLadderName</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>GradesInProgressionLadder</td>
<td>2010/01/01</td>
<td>1</td>
<td>COMMON</td>
<td>STAFF1</td>
<td>GSP Staff</td>
</tr>
<tr>
<td>MERGE</td>
<td>GradesInProgressionLadder</td>
<td>2010/01/01</td>
<td>2</td>
<td>COMMON</td>
<td>STAFF2</td>
<td>GSP Staff</td>
</tr>
<tr>
<td>MERGE</td>
<td>GradesInProgressionLadder</td>
<td>2010/01/01</td>
<td>3</td>
<td>COMMON</td>
<td>STAFF3</td>
<td>GSP Staff</td>
</tr>
<tr>
<td>MERGE</td>
<td>GradesInProgressionLadder</td>
<td>2010/01/01</td>
<td>4</td>
<td>COMMON</td>
<td>STAFF4</td>
<td>GSP Staff</td>
</tr>
</tbody>
</table>

Deleting Progression Grade Ladders Without Steps

This example ProgressionGradeLadder.dat file deletes a progression grade ladder and references it by its source key. This progression grade ladder includes grades without steps. Therefore, you must delete any associated Progression Grade Rate and Progression Rule objects separately.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>ProgressionGradeLadder</th>
<th>SourceSystemOwner</th>
<th>SourceSystemId</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE</td>
<td>ProgressionGradeLadder</td>
<td>VISION</td>
<td>GSP_DEVELOPMENT</td>
</tr>
</tbody>
</table>

Guidelines for Loading Progression Grade Rates

A progression grade rate is a grade rate associated with a progression grade ladder that doesn't use grade steps. The progression grade rate identifies the currency and frequency of payment for its associated grade rate values. The grade rate values specify the salary ranges and rate values for each grade in the ladder. This topic describes some aspects of progression grade rates that you must understand to load them successfully using HCM Data Loader.

Preparing to Load Progression Grade Rates

Before you load progression grade rates, these referenced objects must exist in the target environment:

- Grade and Grade Set
- Legislative Data Group
- Progression Grade Ladder

Progression Grade Rate Names

When you create a progression grade ladder on the Manage Progression Grade Ladders page, the associated progression grade rate is created automatically. It has the same name as the progression grade ladder. Only one progression grade rate can exist for a given progression grade ladder, and both objects have the same legislative data group. When you create a progression grade rate using HCM Data Loader, give it the same name and LDG as its associated progression grade ladder.

Progression Grade Rates in the Application

No independent user interface exists for creating and maintaining Progression Grade Rate objects. You create and maintain progression grade rates in the application using the Manage Progression Grade Ladders page. You can also view progression grade rates on the Manage Grade Rates page, as progression grade rates and grade rates share the same underlying tables. However, you must not create progression grade rates on the Manage Grade Rates page. This restriction exists because the values required for use in grade step progression wouldn't be populated. You can either create...
progression grade rates automatically, when you create a progression grade ladder in the application, or load them using HCM Data Loader.

Examples of Loading Progression Grade Rates

This topic provides examples showing how to load Progression Grade Rate objects using HCM Data Loader.

Creating Progression Grade Rates

In both examples in this section, the progression grade rate:

- Uses grades without grade steps, and the rates are quoted as monthly amounts
- Has the same name as its associated progression grade ladder
- Has a RateType of SALARY, and the ProgressionRateFlag is set to Y to indicate that this rate is to be used for grade step progression

This example ProgressionGradeRate.dat file creates a Progression Grade Rate object using source keys.

```
METADATA|ProgressionGradeRate|SourceSystemOwner|SourceSystemId|EffectiveStartDate|GradeRateName|
LegislativeDataGroupId(SourceSystemId)|RateType|CurrencyCode|RateFrequency|ActiveStatus|AnnualizationFactor|
ProgressionRateFlag|GradeLadderId(SourceSystemId)
MERGE|ProgressionGradeRate|VISION|GSP_DEVELOPMENT_GR|2000/02/15|GSP Development|GSP_US_LDG1|SALARY|USD|MONTHLY|A|12|Y|GSP DEVELOPMENT
METADATA|ProgressionGradeRateValue|SourceSystemOwner|SourceSystemId|EffectiveStartDate|
LegislativeDataGroupId(SourceSystemId)|RateId(SourceSystemId)|RateObjectId(SourceSystemId)|MinimumAmount|MaximumAmount|ValueAmount|GradeRateName|SetCode|GradeCode|LegislativeDataGroup
MERGE|ProgressionGradeRateValue|VISION|GSP_DEVELOPMENT_GRV1|2000/02/15|GSP_US_LDG1|GSP_DEVELOPMENT_GR|DEV1_GSP|3500|6500|5000
MERGE|ProgressionGradeRateValue|VISION|GSP_DEVELOPMENT_GRV2|2000/02/15|GSP_US_LDG1|GSP_DEVELOPMENT_GR|DEV2_GSP|4500|7500|6000
MERGE|ProgressionGradeRateValue|VISION|GSP_DEVELOPMENT_GRV3|2000/02/15|GSP_US_LDG1|GSP_DEVELOPMENT_GR|DEV3_GSP|5500|8500|7000
MERGE|ProgressionGradeRateValue|VISION|GSP_DEVELOPMENT_GRV4|2000/02/15|GSP_US_LDG1|GSP_DEVELOPMENT_GR|DEV4_GSP|6500|9500|8000
MERGE|ProgressionGradeRateValue|VISION|GSP_DEVELOPMENT_GRV5|2000/02/15|GSP_US_LDG1|GSP_DEVELOPMENT_GR|DEV5_GSP|7500|10500|9000
MERGE|ProgressionGradeRateValue|VISION|GSP_DEVELOPMENT_GRV6|2000/02/15|GSP_US_LDG1|GSP_DEVELOPMENT_GR|DEV6_GSP|8500|11500|10000
```

This example ProgressionGradeRate.dat file creates a Progression Grade Rate object using user keys.

```
METADATA|ProgressionGradeRate|EffectiveStartDate|RateType|GradeRateName|GradeLadderName|CurrencyCode|
RateFrequency|AnnualizationFactor|ActiveStatus|LegislativeDataGroup|ProgressionRateFlag
MERGE|ProgressionGradeRate|2010/01/01|Salary|GSP Staff|GSP Staff|USD|Monthly|12|A|GBI United States LDG|Y
METADATA|ProgressionGradeRateValue|EffectiveStartDate|MinimumAmount|MaximumAmount|MidValueAmount|ValueAmount|GradeRateName|SetCode|GradeCode|LegislativeDataGroup
MERGE|ProgressionGradeRateValue|2010/01/01|2000|5000|3500|3000|GSP Staff|COMMON|STAFF1|GBI United States LDG
MERGE|ProgressionGradeRateValue|2010/01/01|4000|6000|5000|4500|GSP Staff|COMMON|STAFF2|GBI United States LDG
MERGE|ProgressionGradeRateValue|2010/01/01|5000|8000|6500|5800|GSP Staff|COMMON|STAFF3|GBI United States LDG
MERGE|ProgressionGradeRateValue|2010/01/01|6000|9000|7500|6200|GSP Staff|COMMON|STAFF4|GBI United States LDG
```
Deleting Progression Grade Rates

When you delete a progression grade rate, its grade rate values are also deleted. You can delete a progression grade rate using HCM Data Loader provided that another object, such as a salary basis, isn’t referencing it. If the progression grade rate is used in a progression grade ladder, then delete it only if you’re also deleting the related progression grade ladder.

This example ProgressionGradeRate.dat file deletes a progression grade rate, which is identified using source keys.

```
METADATA|ProgressionGradeRate|SourceSystemOwner|SourceSystemId|LegislativeDataGroupId(SourceSystemId)
DELETE|ProgressionGradeRate|VISION|GSP_DEVELOPMENT_GR|GSP_US_LDG1
```

This example ProgressionGradeRate.dat file deletes a progression grade rate, which is identified using user keys.

```
METADATA|ProgressionGradeRate|LegislativeDataGroup|GradeRateName
DELETE|ProgressionGradeRate|GBI United States LDG|GSP Staff
```

Guidelines for Loading Progression Rules

A progression rule defines the conditions that determine whether a worker can move to the next grade or grade step in a progression grade ladder. If you’re using grade step progression for your progression grade ladder, then you load the progression rules using the Progression Rule object. This topic describes some aspects of progression rules that you must understand to load them successfully using HCM Data Loader.

Preparing to Load Progression Rules

Before you can load Progression Rule objects, these referenced objects must exist in the target environment:

- Progression Grade Ladder
- Eligibility Profile

Progression Rules and Eligibility Profiles

A progression rule is simply the name of an eligibility profile. The eligibility profile defines the conditions that must be met for progression to occur. You can associate a progression rule with a progression grade ladder, grade, or grade step. The Progression Rule object loads only the association between the eligibility profile and the progression grade ladder, grade, or grade step. Multiple progression rules can exist at each level.

Eligibility profiles associated with Progression Rule objects:

- Must have a usage of Compensation or Global
- Can’t be referenced from the Progression Rule object using source keys
Examples of Loading Progression Rules

This topic provides examples showing how to load Progression Rule objects using HCM Data Loader.

Creating Progression Rules

Typically, you load progression rules for either each step of the progression grade ladder or each grade if your grades don’t have steps. These examples show how to load progression rules for the progression grade ladder, grade, and grade step. They include the attributes necessary to build the keys at each level. These rules apply to this approach:

- When loading a rule for the progression grade ladder, set the SetCode, Gradeld, and GradeStepId or GradeStepName attributes to #NULL.

- When loading a rule for a particular grade, set the GradeStepId or GradeStepName attribute to #NULL.

- When loading a rule for a step, supply values for all attributes.

- If your grades don’t have steps, then remove the GradeStepId or GradeStepName attribute from your file.

Alternatively, you can load the progression rules for each level separately. In this case, you include only the necessary key attributes. For example, when loading progression rules for only the progression grade ladder, you remove the SetCode, Gradeld, and GradeStepId or GradeStepName attributes. Similarly, when loading progression rules for only the grades, you remove the GradeStepId or GradeStepName attribute.

This example ProgressionRule.dat file creates progression rules at each level using source keys. Existing eligibility profiles define the criteria for progression.

// Note: You can’t reference the eligibility profiles using source keys.

METADATA|ProgressionRule|SourceSystemOwner|SourceSystemId|EffectiveStartDate|GradeLadderId(SourceSystemId)|SetCode|Gradeld(SourceSystemId)|GradeStepId(SourceSystemId)|ProgressionRuleName
COMMENT Rule for Progression Grade Ladder
MERGE|ProgressionRule|VISION|GSP_ANALYSTS_PGL_RULE1|2000/02/15|GSP_ANALYSTS_PGL|#NULL|#NULL|#NULL|Active Employees
COMMENT Rule for Grade
MERGE|ProgressionRule|VISION|GSP_ANALYSTS_PGL_RULE_G1_1|2000/02/15|GSP_ANALYSTS_PGL|COMMON|ANALYST1_GR|#NULL|26 Weeks of Service
COMMENT Rules for Steps
MERGE|ProgressionRule|VISION|GSP_ANALYSTS_PGL.Rule_G1S1_1|2000/02/15|GSP_ANALYSTS_PGL.COMMON|ANALYST1_GR|ANALYST1_GRS1|Completed Training
MERGE|ProgressionRule|VISION|GSP_ANALYSTS_PGL.Rule_G1S1_2|2000/02/15|GSP_ANALYSTS_PGL.COMMON|ANALYST1_GR|ANALYST1_GRS1|Completed Training Level 2

This example ProgressionRule.dat file creates progression rules at each level using user keys. Existing eligibility profiles define the criteria for progression.

METADATA|ProgressionRule|GradeLadderName|SetCode|GradeCode|GradeStepName|ProgressionRuleName|EffectiveStartDate
COMMENT Rule for Progression Grade Ladder
MERGE|ProgressionRule|GSP Nurses|#NULL|#NULL|#NULL|Active Employees|2016/01/01
COMMENT Rules for Grade
MERGE|ProgressionRule|GSP Nurses|COMMON|NURSE_1|#NULL|26 Weeks of Service|2016/01/01
COMMENT Rules for Steps
MERGE|ProgressionRule|GSP Nurses|COMMON|NURSE_1|Step 1|Completed Training|2016/01/01
MERGE|ProgressionRule|GSP Nurses|COMMON|NURSE_1|Step 1|Completed Training Level 2|2016/01/01

ORACLE
Deleting Progression Rules

You can delete a progression rule record using HCM Data Loader. Deleting a progression rule doesn't delete the associated eligibility profiles.

This example ProgressionRule.dat file deletes three progression rules. It references the progression rules using source keys.

```plaintext
METADATA|ProgressionRule|SourceSystemOwner|SourceSystemId
COMMENT Delete Rule for Progression Grade Ladder
DELETE|ProgressionRule|VISION|GSP_ANALYSTS_PGL_RULE1
COMMENT Delete Rule for Grade
DELETE|ProgressionRule|VISION|GSP_ANALYSTS_PGL_RULE_G1_1
COMMENT Delete Rule for Step
DELETE|ProgressionRule|VISION|GSP_ANALYSTS_PGL_RULE_G1S1_1
```

This example ProgressionRule.dat file also deletes three progression rules. It references the progression rules using user keys. These rules apply when you delete rules at all levels in a single ProgressionRule.dat file:

- When deleting a rule for the progression grade ladder, set the SetCode, GradeCode, and GradeStepName attributes to #NULL.
- When deleting a rule for a grade, set the GradeStepName attribute to #NULL.
- When deleting a rule for a step, supply values for all attributes.
- If your grades don't have steps, then remove the GradeStepName attribute from your file.

```plaintext
METADATA|ProgressionRule|GradeLadderName|SetCode|GradeCode|GradeStepName|ProgressionRuleName
COMMENT Delete Rule for Ladder
DELETE|ProgressionRule|GSP Nurses|#NULL|#NULL|#NULL|Active Employees
COMMENT Delete Rule for Grade
DELETE|ProgressionRule|GSP Nurses|COMMON|NURSE_1|#NULL|26 Weeks of Service
COMMENT Delete Rule for Step
DELETE|ProgressionRule|GSP Nurses|COMMON|NURSE_1|Step 1|Completed Training
```

Guidelines for Loading Salary Basis Records

This topic describes aspects of the Salary Basis object that you must understand to load salaries and salary bases successfully using HCM Data Loader.

Payroll Elements

Elements that you link to salary bases must exist in the target environment before you load salary bases. In addition, you must create those elements as recurring elements:

- In the same legislative data group as the salary basis
- With a primary classification of either Standard Earnings or Information
- With open eligibility, meaning that no eligibility criteria are defined at the element level

To link the same payroll element to multiple salary bases, you must configure the element to support multiple entries in the same period.

If any of these conditions for payroll elements isn't met, then your salary bases may not load successfully. For example, they may not be associated with a legislative data group or linked with a payroll element. In this case, you correct the salary bases...
Grade Rates
You can link a grade rate to a salary basis to validate worker salaries against the salary ranges defined for the grade. These rules apply:

- The worker’s grade must be included in the grade rate linked to the worker’s salary basis.
- The grade rate currency must match the currency of the payroll element linked to the salary basis.
- The grade rate must be defined in the same legislative data group as the salary basis.

Salary Basis Code
The `SalaryBasisCode` attribute records the frequency at which base pay is stored and displayed. Valid values are defined in the `CMP_SALARY_BASIS` lookup type. The value `PERIOD` (payroll period) means that the salary frequency comes from the payroll defined at the payroll-terms level. When Salary Basis Code is `PERIOD`, leave the `SalaryAnnualizationFactor` attribute blank.

Salary Basis Components
Use the `ComponentUsage` attribute of the Salary Basis object to specify whether you’re loading salary basis components, which itemize salary adjustments. Valid values are defined in the `CMP_COMPONENT_USAGE` lookup type and shown in this table.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO_COMPONENT</td>
<td>Components aren’t used and aren’t loaded with the Salary Basis object.</td>
</tr>
<tr>
<td>SELECTED_COMPONENTS</td>
<td>Components are used and are loaded with the Salary Basis object.</td>
</tr>
<tr>
<td>USER_DECIDES_USAGE</td>
<td>Managers select the components to use when adjusting salary. Components aren’t loaded with the Salary Basis object.</td>
</tr>
</tbody>
</table>

If you set the `ComponentUsage` attribute to `SELECTED_COMPONENTS`, then you load the components using the `SalaryBasisComponent` discriminator. If you omit the `ComponentUsage` attribute, then `NO_COMPONENT` is assumed.

Tip: If you’re unsure about component usage, then specify `USER_DECIDES_USAGE`. This value is the least restrictive.

Provide one Salary Basis Component record for each component type. The `AttributeValue` attribute of the Salary Basis Component record identifies the component type, such as `PROMOTIONAL`, `COST_OF_LIVING`, or `ADJUSTMENT`. This value is validated against the `CMP_SALARY_COMPONENTS` lookup type.

Deletion of Salary Basis Records
You can delete a Salary Basis object using HCM Data Loader only if it’s unassigned and hasn’t been used to create salary records. You can delete individual Salary Basis objects most easily in the application.
Related Topics
• Configuring Payroll Elements for Use in Oracle Fusion Compensation (1589502.1)

Guidelines for Loading Salary Records
A salary record holds a worker’s base pay amount for a specified period and currency. It also displays the salary range used to validate base pay and calculate metrics such as *compa-ratio* and *quartile*. Depending on how the associated *salary basis* is configured, salary can include zero or more components. This topic describes aspects of the *Salary* object that you must understand to load salary records successfully using HCM Data Loader.

Before You Load Salary Records
You must make sure of these configuration conditions before you can load *Salary* objects:

- Employment records must exist in the target environment.
- Worker assignments must include a *grade*, and the grade must be included in the *grade rate* that’s linked to each worker’s salary basis. These requirements must be met for salary range to appear and salary metrics to be calculated correctly on the salary record.

Workers whose salary basis has a frequency of *PERIOD* must be assigned to a payroll defined at the payroll terms level. This payroll assignment must be complete before you load salary records.

Load Salary
Supply salary data in the *Salary.dat* file for processing by HCM Data Loader. If you’re loading *salary components*:

- Set the *MultipleComponents* attribute of the *Salary* object to *Y*. Otherwise, set it to *N*.
- Set the *SalaryAmount* attribute of the *Salary* object to the base salary amount plus the sum of any salary components.

For terminated assignments, set the *Salary* object *Date To* attribute only when the salary to date is earlier than the assignment end date. Otherwise, the salary end date is set to the assignment end date, and not the date in the associated element.

Set the *ActionCode* attribute of the Salary object to the action associated with the salary record. This table shows valid values for the *ActionCode* attribute.

<table>
<thead>
<tr>
<th>Action Code</th>
<th>Action Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOCATE_GRP_CMP</td>
<td>Allocate Workforce Compensation</td>
</tr>
<tr>
<td>CHANGE_SALARY</td>
<td>Change Salary</td>
</tr>
<tr>
<td>GLB_TRANSFER</td>
<td>Global Transfer</td>
</tr>
<tr>
<td>HIRE</td>
<td>Hire an Employee</td>
</tr>
<tr>
<td>HIRE_ADD_WORK_RELATION</td>
<td>Add Employee Work Relationship</td>
</tr>
</tbody>
</table>
You can create salary at the assignment level, supply a unique reference to the assignment. Grade and FTE are obtained from each assignment.

FTE is determined by comparing the standard working hours of the worker’s legal employer with the working hours on the worker’s assignment. When the payroll element attached to the worker’s salary basis has a primary classification of Standard Earnings, the annual amount sent in the element entry is adjusted for FTE. Otherwise, the amount sent in the element entry isn’t adjusted for FTE.

### Loading Salary Components

Set the **ComponentReasonCode** attribute. Valid values are defined in the CMP_SALARY_COMPONENTS lookup type. Use the Manage Base Pay Lookups task in the Setup and Maintenance work area to manage the lookup type. Each component reason code can be used once only for each salary record.

Specify either a **ChangeAmount** or a **ChangePercentage** attribute value for each component record. Don’t specify both. You add the **ChangeAmount** value to the base pay amount to derive the new salary amount for the salary record. The **ChangePercentage** value is a percentage of the base pay amount. The corresponding percentage value is calculated automatically.

### Updating Existing Salary Records

When you update an open-ended salary, the existing salary is automatically end-dated on the day prior to the from date attribute value supplied for the new salary.

### Examples of Loading Salary Records

This topic shows how to load and manage salary records for a worker assignment using HCM Data Loader.

### Creating a Salary Using Source Keys

This example Salary.dat file creates a Salary object with components against an existing assignment. Source keys are used to identify both the Salary object and the associated assignment.

```
MERGE|Salary|VISION|1012SAL15|1012_EMP_ASG|2015/01/01|65000|SalBasis_1007|CHANGE_SALARY|Y|N
```

```
MERGE|SalaryComponent|VISION|1012SAL15_COL|1012SAL15|2015/01/01|COST_OF_LIVING|60000|Y
```
**Oracle Human Capital Management Cloud**

**Integrating with HCM**

**Chapter 12**

**Loading Compensation Objects**

---

**Updating a Salary Using Source Keys**

This example Salary.dat file updates an existing, open-ended salary record by supplying changes to the Salary object and its components. Source keys are used to identify all components of the Salary object.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>Salary</th>
<th>SourceSystemOwner</th>
<th>SourceSystemId</th>
<th>AssignmentId(SourceSystemId)</th>
<th>DateFrom</th>
<th>DateTo</th>
<th>SalaryAmount</th>
<th>SalaryBasisName</th>
<th>ActionCode</th>
<th>SalaryApproved</th>
<th>MultipleComponents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>Salary</td>
<td>VISION</td>
<td>1012SAL16</td>
<td>1012_EMP_ASG</td>
<td>2016/01/01</td>
<td></td>
<td>70000</td>
<td>SalBasis_1007</td>
<td>CHANGE_SALARY</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>

**Creating a Salary Using User Keys**

This example Salary.dat file creates a Salary object with components against an assignment. User keys are used to identify both the Salary object and the associated assignment.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>Salary</th>
<th>AssignmentNumber</th>
<th>DateFrom</th>
<th>SalaryAmount</th>
<th>SalaryBasisName</th>
<th>ActionCode</th>
<th>SalaryApproved</th>
<th>MultipleComponents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>Salary</td>
<td>1014</td>
<td>2015/01/01</td>
<td>53500</td>
<td>SalBasis_1007</td>
<td>CHANGE_SALARY</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>

**Updating a Salary Using User Keys**

This example Salary.dat file updates an existing, open-ended salary record by supplying changes to the Salary object and its components. User keys are used to identify all components of the Salary object.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>Salary</th>
<th>AssignmentNumber</th>
<th>DateFrom</th>
<th>DateTo</th>
<th>SalaryAmount</th>
<th>SalaryBasisName</th>
<th>ActionCode</th>
<th>SalaryApproved</th>
<th>MultipleComponents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>Salary</td>
<td>1014</td>
<td>2016/01/01</td>
<td></td>
<td>53900</td>
<td>SalBasis_1007</td>
<td>CHANGE_SALARY</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>

**Deleting a Salary**

Use HCM Data Loader to delete salary records only if many records were created or loaded incorrectly and must be deleted in bulk. Unless you’re deleting all salary records for an assignment, only the salary record with the latest salary start date can be deleted. For example, if a salary record has salary start dates of January 1, March 1, and September 1 then you can delete just the September 1 record. Alternatively, you can delete all three records.

This example Salary.dat file deletes an existing salary record. The Salary object is identified by its source key.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>Salary</th>
<th>SourceSystemOwner</th>
<th>SourceSystemId</th>
<th>AssignmentId(SourceSystemId)</th>
<th>DateFrom</th>
</tr>
</thead>
</table>
Examples of Loading Salary Range Differential Profiles and Values

Salary range differential values are multipliers applied to grade rates to adjust them on the basis of worker location. For example, a differential value of 1.2 could be applied in one location while a differential value of 0.80 could be applied in another. These values vary to reflect the cost of living in these locations or the relative ease of hiring, for example. Salary range differential values are associated with a salary range differential profile, which you link to the salary basis. This topic provides examples showing how to load differential profiles and their differential values using HCM Data Loader.

Creating Salary Range Differential Profiles and Values

To create a differential profile, you load the Salary Range Differential object and one or more Salary Range Differential Value components. Each Salary Range Differential Value component is for a single location. Differential profiles reference legislative data groups, and their differential values reference locations. Referenced legislative data groups and locations must exist in the target environment before you load Salary Range Differential objects.

This example SalaryRangeDifferential.dat file creates a differential profile with a single differential value. Both components are identified using source keys.

```
METADATA|SalaryRangeDifferential|EffectiveStartDate|Name|Code|Status|Criteria|SourceSystemId|
SourceSystemOwner|LegislativeDataGroupName
MERGE|SalaryRangeDifferential|2000/01/01|SalDiffName333|SalDiffCode333|ACTIVE|LOCATION|
SalRangeDiffSSID_333|VISION|PM DE LDG
METADATA|SalaryRangeDifferentialValue|EffectiveStartDate|SourceSystemId|SourceSystemOwner|
RangeDiffId(SourceSystemId)|LocationCode|LocationSetCode|Differential
MERGE|SalaryRangeDifferentialValue|2000/01/01|SalDiffCode333|Atlanta|Vision Corporation 202|0.2
```

This example SalaryRangeDifferential.dat file creates a differential profile with a single differential value. Both components are identified using user keys.

```
METADATA|SalaryRangeDifferential|EffectiveStartDate|Name|Code|Status|Criteria|LegislativeDataGroupName
MERGE|SalaryRangeDifferential|2000/01/01|SalDiffName444|SalDiffCode444|ACTIVE|LOCATION|PM DE LDG
METADATA|SalaryRangeDifferentialValue|EffectiveStartDate|RangeDiffCode|LocationCode|LocationSetCode|Differential
MERGE|SalaryRangeDifferentialValue|2000/01/01|SalDiffCode444|Atlanta|Vision Corporation 202|0.2
```
Guidelines for Making Date-Effective Changes to Salary Range Differentials

The Salary Range Differential object is date-effective. That is, a history of changes to the Salary Range Differential object and its child components is maintained and can be viewed. For example, users can see the date-effective history of a differential profile in the Compensation work area. This topic describes how date-effective records are managed for Salary Range Differential objects when you use HCM Data Loader.

Creating Date-Effective Records

When you create or update a differential value, a date-effective record is added to the differential value. A date-effective record must also be added to the parent differential profile if it has no date-effective record for that effective start date. In this case, you must include the parent Salary Range Differential component in the .dat file. It must have the same effective start date as the child differential value.

This table shows a history of changes to a differential profile.

<table>
<thead>
<tr>
<th>RangeDiffId</th>
<th>EffectiveStartDate</th>
<th>EffectiveEndDate</th>
<th>Status</th>
<th>Criteria</th>
<th>LegislativeDataGrCode</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>01-01-1951</td>
<td>01-05-2016</td>
<td>Active</td>
<td>Location</td>
<td>US1</td>
<td>US SRD US Range</td>
</tr>
<tr>
<td>123</td>
<td>02-05-2016</td>
<td>31-12-4712</td>
<td>Active</td>
<td>Location</td>
<td>US1</td>
<td>US SRD US Range</td>
</tr>
</tbody>
</table>

This table shows the related changes to its differential values.

<table>
<thead>
<tr>
<th>RangeDiffValueId</th>
<th>EffectiveStartDate</th>
<th>EffectiveEndDate</th>
<th>Differential</th>
<th>LocationId</th>
</tr>
</thead>
<tbody>
<tr>
<td>234</td>
<td>01-01-1951</td>
<td>31-12-4712</td>
<td>1.3</td>
<td>A</td>
</tr>
<tr>
<td>345</td>
<td>02-05-2016</td>
<td>31-12-4712</td>
<td>0.8</td>
<td>B</td>
</tr>
<tr>
<td>456</td>
<td>02-05-2016</td>
<td>31-12-4712</td>
<td>0.9</td>
<td>C</td>
</tr>
</tbody>
</table>

When the location B value was created, a date-effective record with the location B effective start date was also added to the differential profile. However, when the location C value was created, no date-effective record was added to the differential profile. A record for that effective start date already existed in the differential profile, so no additional record was needed. If you create or update multiple differential values on the same date, then only one date-effective record is added to the parent differential profile.

**Note:** You don’t have to include the parent Salary Range Differential component in the .dat file if it already has a date-effective record with the required effective start date.
Editing the Salary Range Differential Component

If you correct or update the parent Salary Range Differential object, then no date-effective record is added to its child components. For example, if you make the differential profile inactive, then a new date-effective record is added to the parent component. However, no date-effective record is added to the differential values.
13 **Loading Payroll Relationships**

**Payroll Information**

**Overview of Loading Payroll Details**

Use HCM Data Loader to load payroll information at the payroll relationship, payroll assignment records, and assigned payroll levels. For example, you can update the overtime period for employees either at their payroll relationship or assignment level. A time period you specify at a lower level, such as assignment level, overrides a period at a higher level, such as payroll relationship.

This figure shows the payroll information for a person at the Payroll Relationship, Assignment, and Assigned Payroll levels.

Consider these examples that explain the various actions you can take using HCM Data Loader.

- You manage Carrie Smith who’s a part-time temporary employee and is on weekly payroll. Starting one month from now, Carrie accepted an offer to become a full-time permanent employee in the same position. You can update Carrie’s assignment record and transfer her to a payroll appropriate for a full-time permanent employee, such as monthly or semimonthly.

- Anthony has two assignment records, one on a weekly payroll and one on a monthly payroll. On 10-Jun-18, you terminated Anthony’s assignment record on the weekly payroll. The termination process automatically set the last standard process date to the end date of the payroll period. Anthony’s termination package specifies that he should
receive compensation payments through the month of June. To ensure that he’s paid on both weekly and monthly payroll through June, you change the LSPD on weekly payroll to 30-Jun-18.

- You wrongly assigned Sophie, who’s a new hire, to weekly payroll instead of monthly payroll. Now, you can delete her weekly payroll record and add the monthly payroll record.

Related Topics
- Payroll Relationships

Overview of Loading Payroll Relationship Details

Use HCM Data Loader to update payroll details for employees at the payroll relationship level. Payroll relationships group a person’s employment assignment records based on the payroll statutory calculation and reporting requirements. For example, you can update the overtime period for employees at their payroll relationship.

When an employee is hired, the application automatically creates the payroll relationship details. A person could have multiple payroll relationships if they’re associated with more than one PSU or more than one payroll relationship type.

Element Duration Dates

The element duration dates at the payroll relationship level control the start and end date of earnings and deductions assigned to the employee’s payroll relationship. Examples of payroll relationship elements include tax and social insurance entries.

See this table for various element duration dates.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Standard Earnings Date</td>
<td>The date when standard earnings start accumulating. For example, the start date of the employee’s record for the payroll statutory unit.</td>
</tr>
<tr>
<td>Last Standard Earnings Date</td>
<td>The date when standard earnings stop accumulating. For example, the termination date of the employee’s record for the payroll statutory unit.</td>
</tr>
<tr>
<td>Last Standard Process Date</td>
<td>The last date of the payroll period in which the termination of the payroll relationship occurs.</td>
</tr>
<tr>
<td>Final Close Date</td>
<td>The last date the payroll run will include elements (which are set to use final close) for processing. This includes regular and supplemental payroll runs</td>
</tr>
</tbody>
</table>

You can’t edit these element duration dates at the payroll relationship level. These dates are populated based on the date information at the assignment level.

For example, when Joe is terminated on 1-Mar-19, the application sets the last standard earnings date on his assignment record with this date.
As the figure shows, the same date cascades up to the last standard earnings date field at the payroll relationship level.

Assigned Payroll

Example of Loading Assigned Payroll Details

Use HCM Data Loader to assign a payroll to an assignment. Though you can specify the Time card Required and Overtime Status details at the assigned payroll level, consider defining them for an assignment.

For example, an employee is assigned to the weekly payroll and at the assigned payroll level, you load the overtime period code. If the employee transfers to the monthly payroll, the overtime period will no longer apply. In this case, you must define the overtime period at the assignment level.

Use HCM Data Loader to manage all payrolls for an assignment and for a given payroll relationship.

For example, in this DAT file, you assign a weekly payroll to an employee.

```
METADATA|AssignedPayroll|EffectiveStartDate|AssignmentNumber|PayrollDefinitionCode|LegislativeDataGroupName|StartDate
MERGE|AssignedPayroll|2019/03/01|E10001|Weekly Payroll|Vision US LDG|2019/03/01
```

Related Topics

- Payroll Relationships
- Element Duration Dates
Example of Loading Element Duration Dates for Assigned Payroll

The element duration dates at assigned payroll level control when the assignment will be processed in a specific payroll run. Let’s consider this example.

John is on a monthly payroll, but starting 20-Feb-18, he’s transferred to a semi-monthly payroll. His monthly and semi-monthly payroll definitions looks like this. You can update the final close date as 31-Jul-18.

<table>
<thead>
<tr>
<th>Time Definition</th>
<th>Monthly Payroll</th>
<th>Semi-Monthly Payroll</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Standard Earnings Date</td>
<td>01/01/18</td>
<td>02/20/2018</td>
</tr>
<tr>
<td>Last Standard Earnings Date</td>
<td>02/19/2018</td>
<td></td>
</tr>
<tr>
<td>Last Standard Process Date</td>
<td>02/28/2018</td>
<td></td>
</tr>
<tr>
<td>Final Close</td>
<td>07/31/2018</td>
<td></td>
</tr>
</tbody>
</table>

This image shows the various time definition dates for monthly and semi-monthly payroll.
Use this dat file to set the final close date as 31-Jul-18.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>AssignedPayroll</th>
<th>EffectiveStartDate</th>
<th>FinalCloseDate</th>
<th>LegislativeDataGroupName</th>
<th>PayrollDefinitionCode</th>
<th>AssignmentNumber</th>
<th>StartDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>AssignedPayroll</td>
<td>2018/01/01</td>
<td>2018/07/31</td>
<td>Vision US LDG</td>
<td>Monthly Payroll</td>
<td>E10003</td>
<td>2018/01/01</td>
</tr>
</tbody>
</table>

Related Topics

- Payroll Relationships
- Element Duration Dates

Payroll Assignments

Example of Loading Element Duration Dates For Payroll Assignments

Use HCM Data Loader to load payroll related information for an assignment, such as element duration dates. The element duration dates at assignment level control the start and end date of earnings and deductions assigned to the employee’s assignment, such as salary.

Example

On 13-Feb-2019, Maria Peterson who’s on weekly payroll terminates her service with your company. The termination process automatically sets these process dates:

- The last standard earnings date to 13-Feb-19.
- The last standard process date to the end date of her weekly payroll, which is 15-Feb-19.

However, Maria is still eligible to receive outstanding overtime and commission payments, which can be paid up to 6 months in arrears. To ensure that she’s paid these payments, you can update the final close date on the assignment record to 31-Dec-19.
As this image shows, the termination process sets the process dates for Maria.

**Termination Date**

- **13-Feb**
- **15-Feb**
- **31-Dec**

**Maria is eligible to receive payments**

- **LSPD = 15-Feb**
- **LSED = 13-Feb**
- **FC = 31-Dec**

In this dat file, you update the final close date to 31-Dec-2019.

```
METADATA|ElementDurationDate|DateValue|SourceType|AssignmentNumber|TimeDefinitionCode
MERGE|ElementDurationDate|2019/12/31|PA|E10001|Final Close
```

**Update the Time Card Required Option for Assignments**

You can use the **Time Card Required** option to indicate if the assignment is paid by time card or not. At the assignment level, the time card status applies to the specific assignment record.
Let’s consider this example. Tina is on a temporary contract. She submits her time cards every week and receives her pay checks on a weekly basis. Effective 15-July, the company offers her a permanent contract. Under the new contract, she will become a salaried employee and will be paid on a monthly basis. And she will no longer need to submit the time card.

Use this dat file to set the **Time Card Required** parameter to **No**.

```sql
METADATA|PayrollAssignmentDetails|EffectiveStartDate|TimecardRequiredFlag|AssignmentNumber
MERGE|PayrollAssignmentDetails|2019/07/15|N|E10002
```

### Update the Last Standard Process Date for Assignments

You can sequence the loading of the element duration dates depending on whether you move the element duration dates forward or backward.

This table describes the various actions involved in updating the last standard process date.

<table>
<thead>
<tr>
<th>Action</th>
<th>What You Do</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move the element duration dates forward.</td>
<td>Load the final close date first.</td>
<td>You update the final close date from 15-Mar to 15-May and the last standard process date from 15-Feb to 15-Apr.</td>
</tr>
<tr>
<td>Move the element duration dates backward.</td>
<td>Load the last standard process date first.</td>
<td>You update the last standard process date from 15-APR to 15-FEB and the final close date from 15-May to 15-Mar.</td>
</tr>
</tbody>
</table>

Let’s consider this scenario where you hire an employee on 1-Jan-15 and create their Tax withholding calculation card. You terminate the employee on 15-Jan-16. Further, you update the final close date as 15-Mar-16 and the last standard process date as 15-Feb-16 at the assignment level. At a later point in time, you update the LSPD as 15-Apr-16 and FC as 15-May-16.

You must use these two DAT files to load the LSPD and FC date values separately.
<table>
<thead>
<tr>
<th>METADATA</th>
<th>ElementDurationDate</th>
<th>DateValue</th>
<th>SourceType</th>
<th>AssignmentNumber</th>
<th>TimeDefinitionCode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Final Close</td>
<td>2016/05/15</td>
<td>PA</td>
<td>E10001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TimeDefinitionCode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Last Standard Process Date</td>
<td>2016/04/15</td>
<td>PA</td>
<td>E10001</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Examples of Loading Legislative Data Groups

Legislative data groups are used to partition payroll and related data. At least one legislative data group is required for each country where the enterprise operates. Each legislative data group holds a legislation code, currency, and cost allocation key flexfield. It's also associated with one or more payroll statutory units. This topic provides examples showing how to load and manage Legislative Data Group objects using HCM Data Loader.

Creating Legislative Data Groups

This example LegislativeDataGroup.dat file creates a legislative data group for the United Kingdom. It identifies the legislative data group using its source key. Codes are supplied for the currency and legislation.

```
METADATA|LegislativeDataGroup|SourceSystemOwner|SourceSystemId|Name|LegislationCode|DefaultCurrencyCode
MERGE|LegislativeDataGroup|VISION|LDG_VI_UK|Vision UK|GB|GBP
```

This example LegislativeDataGroup.dat file creates a legislative data group for the United States and associates it with a cost allocation key flexfield.

```
METADATA|LegislativeDataGroup|Name|Territory|DefaultCurrency|StructureInstanceName
MERGE|LegislativeDataGroup|Vision US|United States|US Dollar|System Test Costing
```

Deleting Legislative Data Groups

You can delete Legislative Data Group objects using HCM Data Loader. This example LegislativeDataGroup.dat file deletes a legislative data group. It identifies the legislative data group by its source key.

```
METADATA|LegislativeDataGroup|SourceSystemOwner|SourceSystemId
DELETE|LegislativeDataGroup|VISION|LDG_VI_UK
```

Object Groups

Overview

Use HCM Data Loader to define object groups that define subsets of people or elements for processing or reporting.

Define one of these object groups.

- Element Group
- Payroll Relationship Group
- Process Information Group
Element Group

An element group defines a group of individual elements for processing payroll runs or reports. Element groups are static, which means you select each element manually to include in a group.

Let's consider this example. In an off-cycle payroll run, you want to make expense payments that are not subject to tax and other deductions. You create an element group for all expense elements. And you submit a payroll run with this element group as the parameter.

Use this .dat file to create an element group for all your required expenses.

```
METADATA|ObjectGroup|LegislativeDataGroupName|ObjectGroupCode|ObjectGroupName|ObjectGroupTypeCode|StartDate|EndDate|StaticDynamicFlag|Description
MERGE|ObjectGroup|US LDG|VIS_EXP_GROUP|EXPENSE_ELEMENT_GRP|ELEGRP|2010/01/01||N|Expense elements group for off-cycle payments.
METADATA|ObjectGroupAmend|LegislativeDataGroupName|ObjectGroupCode|ObjectGroupLevelName|ObjectCode|
IncludeOrExclude
MERGE|ObjectGroupAmend|US LDG|VIS_EXP_GROUP|Element|Travel_Allowance|I
MERGE|ObjectGroupAmend|US LDG|VIS_EXP_GROUP|Element|Per_Diem|I
```

Payroll Relationship Group

Use a payroll relationship group to limit the persons that you want to process for payroll, data entry, and reporting. You can define your payroll relationship group to be either static or dynamic. In a dynamic group, you select a formula that determines the people to include in the group. You can use either the Payroll Relationship Number or Assignment Number or both to create the group.

In this example, Vision Corporation wants to pay bonus to a group of 25 employees in an off-cycle payroll run. Here, you create a payroll relationship group for these employees. When submitting the payroll run, you select the payroll relationship group as the parameter.

Use this .dat file to create a payroll relationship group.

```
METADATA|ObjectGroup|LegislativeDataGroupName|ObjectGroupCode|ObjectGroupName|ObjectGroupTypeCode|StartDate|EndDate|StaticDynamicFlag|Description
METADATA|ObjectGroupAmend|LegislativeDataGroupName|ObjectGroupCode|ObjectGroupLevelName|ObjectCode|
IncludeOrExclude
MERGE|ObjectGroupAmend|US LDG|VIS_BONUS0919|Payroll Relationship|8186051|I
MERGE|ObjectGroupAmend|US LDG|VIS_BONUS0919|Payroll Relationship|8186965|I
MERGE|ObjectGroupAmend|US LDG|VIS_BONUS0919|Payroll Relationship|8186043|I
MERGE|ObjectGroupAmend|US LDG|VIS_BONUS0919|Payroll Relationship|8187231|I
MERGE|ObjectGroupAmend|US LDG|VIS_BONUS0919|Payroll Relationship|8181098|I
```

Related Topics

- Object Group HCM Data Loader Files for Bank Reprocessing

Example of Loading Object Groups

Rahul has an invalid branch number due to the closure of the bank. And you want to update his bank account details. In this case, you don’t end date the personal payment method, but update the existing personal payment method with the correct bank details.
To update Rahul's branch number, you do these tasks as shown in the figure.

1. Using HCM Data Loader, load the details returned by the bank.
2. Submit the **Process Bank Corrections File** flow to update his personal payment method.

---

### Load the Bank Account Details

The bank returns a file containing details of the payments that are rejected. To process this bank information, use these .dat files.

1. **ObjectGroup.dat file** to create the object group.

   ```
   METADATA|ObjectGroup|SourceSystemOwner|SourceSystemId|LegislativeDataGroupName|ObjectGroupCode|
   ObjectGroupName|ObjectGroupTypeCode|Description|StartDate|EndDate|FormulaName
   MERGE|ObjectGroup|VISION|PPM_E8185311|US LDG|PPM_E8185311|PPM Update for 8185311|PROCINFO|PPM update for employee 8185311|2019/10/01|
   ```

2. **ObjectGroupStore.dat file** to load the bank information returned by the bank.

   ```
   METADATA|ObjectGroupStore|SourceSystemOwner|SourceSystemId|LegislativeDataGroupName|
   ObjectGroupId(SourceSystemId)|SequenceNumber|FLEX:PAY_OBJECT_GROUP_STORE_DDF|
   amount(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   assignmentNumber(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   checkNumber(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   makeExternalPayment(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   paymentDate(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   paymentReference(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   processDate(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   replacementAccountType(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   replacementBranchNumber(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   returnreasoncode(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   employeeName(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   replacementAccountNumber(PAY_OBJECT_GROUP_STORE_DDF=ORA_PAY_BANK_CORRECTION)|
   MERGE|ObjectGroupStore|VISION|E8185311_BANK_CORRECTION|US LDG|PPM_E8185311|1|ORA_PAY_BANK_CORRECTION|
   5000|E8185311|222324|YES|2019/10/31 00:00:00|185805|2019/10/31 00:00:00|Checking|004|AccountType Chg|Rahul|00456754231
   ```
Submit the Process Bank Corrections File

From the Payroll Checklist work area, submit the **Process Bank Corrections File** flow. When you load the ObjectGroupStore.dat file into the Oracle Web Center Content server, a UCM ID is generated. The **Process Bank Corrections File** flow uses the UCM ID to retrieve and use the bank information necessary for the flow. The flow updates the Personal Payment Method with the new branch number details.

**Related Topics**

- Object Group HCM Data Loader Files for Bank Reprocessing

Example of Loading Payroll Consolidation Group

A payroll consolidation group is a grouping of payroll runs within the same time period for the same payroll. For this group, you can run reporting, costing, and post-run processing. For example, a group might consolidate payroll runs for different work sites into one group for payments and costing.

Supply comments in a separate text file and then pass the text file name to the **Comments** attribute. Further, place the comments text file in the **ClobFiles** folder within the same compressed file as the PayrollConsolidationGroup.dat file that references it.

**Note:** Don’t supply the text directly in the PayrollConsolidationGroup.dat file.

Use the following example dat file to load payroll consolidation groups:

```
METADATA|PayrollConsolidationGroup|SourceSystemOwner|SourceSystemId|LegislativeDataGroupId(SourceSystemId)|ConsolidationSetName|Comments
MERGE|PayrollConsolidationGroup|VISION|INFUSION_US_EMP|HDL_LDG_VISION_US|InFusion US Employee Group|
Consolidation group description
```

**Related Topics**

- How Pay Frequency Components Work Together
- Consolidation Groups: Explained_Guide_Only_Topic

Examples of Loading Payroll Definitions and Time Periods

Payroll definitions contain calendar and offset information that determine when to calculate and cost payments. It includes definition of payment frequency, processing schedule, and other parameters for a particular payroll.

For example, to pay employees semimonthly, you can create a payroll definition using the semimonthly payroll period type, ensuring that tax and other calculations produce correct results for those employees.

You can’t change either the first effective start date or last effective end date for an existing payroll definition. Create payroll definition objects with effective start dates on or before the start dates of other objects that refer to your definitions.
In this example, the Vision Corp Company requires payroll definitions for two sets of employees. One set is permanent salaried employees who are paid on a semimonthly basis, and the other is temporary employees that are paid on a monthly basis using time card data.

**Note:** The Company hires all employees after the effective start date of this payroll definition, so there is no issue with loading historical employee data.

The following figure shows the two payroll definitions that indicate the pay periods to process and links the employees with the payroll run.

Before using the HCM Data Loader to create payroll definitions, ensure that you load the following objects:

- Organization Payment Methods for your payrolls. For more information about loading default payment methods, refer to the topic: Loading Organization Payment Methods: Explained.
- Consolidation group that processes the results of one or more payroll runs in a single action. For more information about loading consolidation groups, refer to the topic: Loading Payroll Consolidation Group: Explained.
Payroll Offset

Payroll offsets help you define your payroll cycle schedule. You can select for your payroll cycle events to occur on specific dates or be based on offsets from period start or end dates. Based on your choices and the number of calendar years you specify, the complete calendar for your payroll’s lifetime is generated. This table describes the predefined payroll cycle events that you can offset.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planned Submission Date</td>
<td>The date on which the user will submit the payroll run.</td>
</tr>
<tr>
<td>Cutoff Date</td>
<td>The final date that payroll information can be entered for the payroll period.</td>
</tr>
<tr>
<td>Payslip Availability Date</td>
<td>The date on which payees can view payslips.</td>
</tr>
<tr>
<td>Payroll Run Date</td>
<td>The date the payroll calculation process uses to retrieve effective values, such as employee details.</td>
</tr>
<tr>
<td>Date Earned</td>
<td>The date on which the payroll process processes element entries for the payroll run. The date earned must be within the effective dates of the payroll period.</td>
</tr>
<tr>
<td>Date Paid</td>
<td>The date the employee is marked as paid. For check payments, this is the date that the check is valid for cash or deposit. For electronic funds transfer (EFT) payments, it is the transfer date.</td>
</tr>
</tbody>
</table>

This example does the following:

- Loads the payroll definition that pays employees into their bank accounts on a semimonthly basis.
- Includes dynamically generated offset dates.
- Identifies the default payment method for those employees who have not chosen a personal payment method.
- Adjusts payroll days.

Here, you specify the number of years ahead that the payroll cycle dates will be generated as 5. Also, you load the following offsets to your payroll processing schedule. In this example, the payroll cutoff date falls 5 days before the date earned for the payroll period.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Falls Value</th>
<th>Day Type Value</th>
<th>Offset Value</th>
<th>Base Date Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cutoff Date</td>
<td>5</td>
<td>W</td>
<td>B</td>
<td>E</td>
</tr>
<tr>
<td>Planned Submission Date</td>
<td>4</td>
<td>W</td>
<td>B</td>
<td>E</td>
</tr>
</tbody>
</table>
Load the payroll definition that pays employees by check and by using time card data on a monthly calendar basis.

Load the payroll definition that pays employees by check and by using time card data on a monthly calendar basis.

These payroll definition lines load the following fixed dates to define offsets:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of years</td>
<td>5</td>
</tr>
<tr>
<td>Set Date</td>
<td>Yes</td>
</tr>
<tr>
<td>Cutoff Date</td>
<td>2012/06/25</td>
</tr>
<tr>
<td>Date Earned</td>
<td>2012/06/28</td>
</tr>
<tr>
<td>Payroll Run Date</td>
<td>2012/06/28</td>
</tr>
<tr>
<td>Date Paid</td>
<td>2012/06/28</td>
</tr>
<tr>
<td>Payslip Availability Date</td>
<td>2012/06/28</td>
</tr>
<tr>
<td>Planned Submission Date</td>
<td>2012/06/28</td>
</tr>
</tbody>
</table>

Valid Payment Methods

You can specify an effective date that determines when your organization payment method will be used for this payroll definition. This date range must fit within both the effective date range of the payroll definition and the effective date range of the organization payment method.
Payroll Time Periods

Payroll Time periods represent the various dates within each payroll cycle and their frequency. Payroll time periods hold the payroll definition calendar that is based on the frequency of the payroll, such as weekly and monthly. For example, you can adjust payroll days to account for a bank holiday.

As payroll time periods are generated when the payroll is defined, reference the payroll and time period to adjust using user keys. This is because the source key is generated by Oracle HCM cloud. The following DAT files explain how to adjust a time period for these payroll definitions.

```
METADATA|PayrollTimePeriod|PayrollCode|LegislativeDataGroupName|PeriodCategory|StartDate|PayrollRunDate|DatePaid
```

Related Topics

- Payroll Definitions
- Managing Payroll Definitions
- Creating Payroll Definitions

Example of Loading Time Definitions

Use HCM Data Loader to create time definitions that can be either a date or a span of time. You can define static definitions using multiple periods, additional adjustments, or both. Alternatively, you can also define dates based on dynamic variables. For example, you can create multiple time definitions to establish different workweeks for different facilities or group of employees.

Later, you use these time definitions in payroll periods, balances, overtime periods, balance dimensions, and so on.
This image shows the different types of time definition and how and where you can use them.

### Static Time Definition

Load static time definitions to create recurring patterns of dates, such as overtime periods. These definitions can be static periods of unusual length based on a given date or they can generate dates based on dynamic variables.

**Example**

A company’s overtime period is a standard five day work week. You create a static time definition with a weekly frequency that generates time periods for 3 years.

```
METADATA|PayrollTimeDefinition|TimeDefinitionCode|DefinitionName|StartDate|DayAdjustment|DefinitionType|
NumberOfYears|PeriodType|PeriodUnit|BaseTimeDefinitionCode|DatabaseItemCode|PeriodTimeDefinitionCode|
AdjustmentType|LegislativeDataGroupName MERGE|PayrollTimeDefinition|RPGP_HDL_TD_002|RPGP_HDL_TD_002|2016/01/01||
Static time definition|2|Monthly|||ABC US LDG
```
User-Defined Date Time Definition

You can use these predefined element duration dates to control when element entries for an employee start or end.

<table>
<thead>
<tr>
<th>Type</th>
<th>What Are They</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Standard Earnings Date</td>
<td>The date when standard earnings start accumulating.</td>
<td>Start date of the employee’s record for the payroll statutory unit.</td>
</tr>
<tr>
<td>Last Standard Earnings Date</td>
<td>The date when standard earnings stop accumulating.</td>
<td>Termination date of the employee’s record for the payroll statutory unit.</td>
</tr>
<tr>
<td>Last Standard Process Date</td>
<td>The last day in which the element entries are considered for a normal payroll run.</td>
<td>Last day of the payroll in which termination or payroll transfer occur.</td>
</tr>
<tr>
<td>Final Close Date</td>
<td>The last date the payroll run will include elements for processing.</td>
<td>If a final close date is entered for an assigned payroll record, the employee will no longer be considered for processing by that specific payroll.</td>
</tr>
</tbody>
</table>

Using HCM Data Loader, you can load your own time definitions that define the last standard process and final close dates. For example, Vision Corp provides housing allowance to its employees and wants the allowance to end 30 days after the employee is terminated. You define a user-defined date time definition based on the last standard earnings date, which is the termination date plus 30 days.

Use this PayrollTimeDefinition.dat file to create a user-defined time definition that is 30 days after the last standard earning date.

```
METADATA|PayrollTimeDefinition|TimeDefinitionCode|DefinitionName|StartDate|DayAdjustment|DefinitionType|NumberOfYears|PeriodType|PeriodUnit|BaseTimeDefinitionCode|DatabaseItemCode|PeriodTimeDefinitionCode|AdjustmentType|LegislativeDataGroupName
MERGE|PayrollTimeDefinition|RPGP_HDL_TD_006|RPGP_HDL_TD_006||-2|User-defined date|User Year|First Standard Earning Date|||Subtract Days|ABC US LDG
```

**Time Span Time Definitions**

Use HCM Data Loader to create segments of time that define intervals for scheduled operations, such as payroll frequency. You can use these time span definitions within other time definitions.

Example: A company wants to make employee payments on the last day of the month and third-party payments five days later. The company creates a time definition of type time span and uses it when defining the payment method for the payee. Use this PayrollTimeDefinition .dat file to create a time span time definition.
Retrieval Date Time Definitions

Load retrieval date time definition that is based on the payroll database items of date format.

**Example:**

Let’s assume that the rate contributor for the severance rate definition uses a balance as of termination date. You can define a retrieval date time definition based on a database item that retrieves the termination date. You can then use this time definition as the reference date for the balance value in the rate contributor.

Use this .dat file to create a retrieval date time definition.

**Related Topics**

- Use Time Definitions for Severance Pay

Example of Loading Payroll Element Run Usage

Use the payroll element run usage object to identify how an element is used with the run type. Run types control the elements and payment types to process in a payroll run.

You can’t change either the first effective start date or last effective end date for an existing payroll element run usage object. Create these objects with effective start dates on or before the start dates of other objects that refer to your payroll element run usage.

Vision Corp processes and pays the bonus amounts separately from the regular earnings. The organization pays regular payroll by EFT and issues check bonuses once a year. For this requirement, they create a separate payment for the bonus element entry, which is marked to pay separately. Using HCM Data Loader, the Payroll Manager creates a bonus element, Annual Bonus, with Supplemental Earnings classification. She supplies these details.

<table>
<thead>
<tr>
<th>Element Detail</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Classification</td>
<td>Supplemental Earnings</td>
</tr>
<tr>
<td>Secondary Classification</td>
<td>Bonus</td>
</tr>
<tr>
<td>Category</td>
<td>Standard</td>
</tr>
<tr>
<td>Process and pay element separately or with other earnings elements?</td>
<td>Process separately and pay separately</td>
</tr>
</tbody>
</table>
Further, she loads the run type usage as **Separate Payment** with Element Usage as **Trigger**. The application processes this element as well as makes payments separately.

This example dat file loads the payroll element run usage object:

```
METADATA|PayrollElementRunTypeUsage|EffectiveStartDate|UsageType|ElementTypeCode|RunTypeCode|
LegislativeDataGroupName
MERGE|PayrollElementRunTypeUsage|2018/02/02|Trigger| ANNUAL_BONUS|Regular|Vision Corporation US
```

**Related Topics**
- Elements
- Run Types
- How Pay Frequency Components Work Together

---

**User-Defined Tables**

**Overview**

You can set up your own structured tables to maintain a date effective lists of values or data, such as wage codes. A user-defined table consists of rows and columns and stores the values as cells for a specific row and column combination. For example, you can define a table that shows the bonus amounts in the columns, for the years of services in the rows.

Using HCM Data Loader, you can either create a user-defined table to store a range of values or match a specific value. You can use this table with a formula to store tabular data.

**Rows, Columns, and Values**

To start with, you define the column and row details. Let’s understand what user-defined rows and columns are.

- A table row is a definition that stores row details. For example, a row that stores the years of service details for bonus payments. Define rows to accept either an exact value, such as a grade value, or a range of values, such as a range of salary values.
- A table column stores the column details, such as a column could capture bonus amounts. You can also use a fast formula to validate entries in the user-defined table. For example, you can validate if your bonus entries fall within a specified range.

Then, you load the values for a date-effective combination of a user-defined row and column.

**Related Topics**
- User Table Validation Formula Type
Example of Loading User-Defined Tables for a Range of Row Values

Let's consider this example. Each year, your organization offers stock options to its employees. The amount of stocks depend on the years of service and the job category of the employee receiving them.

As this image shows, the user-defined table contains stock option allocations by job category and years of service. The rows represent a range of years of service during which employees receive the same number of stock options. The columns contain the job categories. And the values represent the number of stock options awarded to the specified job category during the specified years of service.

Use this .dat file to load your stock option allocations.

```
METADATA|UserDefinedTable|UserTableCode|RangeOrMatch|UserKeyUnits|UserRowTitle|UserTableName|LegislativeDataGroupName
METADATA|UserDefinedTableColumn|UserColumnCode|DataType|UserColumnName|UserTableCode|LegislativeDataGroupName|FormulaName
METADATA|UserDefinedTableRow|DisplaySequence|EffectiveEndDate|EffectiveStartDate|RowHighRange|RowLowRangeOrName|UserTableCode|LegislativeDataGroupName|RowName
METADATA|UserDefinedTableColumnInstance|EffectiveEndDate|EffectiveStartDate|Value|LegislativeDataGroupName|UserColumnCode|RowLowRangeOrName|UserTableCode
MERGE|UserDefinedTable|VISION_RANGE_OF_ROW_VALUES|R|N|Years of Service.|VISION_Range_of_Row_Values|US Legislative Data Group
MERGE|UserDefinedTableColumn|EXECUTIVE|N|EXECUTIVE|VISION_RANGE_OF_ROW_VALUES|US Legislative Data Group
MERGE|UserDefinedTableColumn|MANAGER|N|MANAGER|VISION_RANGE_OF_ROW_VALUES|US Legislative Data Group
MERGE|UserDefinedTableColumn|TECHNICAL|N|TECHNICAL|VISION_RANGE_OF_ROW_VALUES|US Legislative Data Group
MERGE|UserDefinedTableColumn|CLERICAL|N|CLERICAL|VISION_RANGE_OF_ROW_VALUES|US Legislative Data Group
MERGE|UserDefinedTableRow|10|4712/12/31|2019/06/14|3|1|VISION_RANGE_OF_ROW_VALUES|US Legislative Data Group
MERGE|UserDefinedTableRow|20|4712/12/31|2019/06/14|5|4|VISION_RANGE_OF_ROW_VALUES|US Legislative Data Group
```
## Loading Payroll Setup Objects

<table>
<thead>
<tr>
<th>UserDefinedTableRow</th>
<th>Date</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Value 3</th>
<th>Value 4</th>
<th>Group</th>
<th>Value 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>8</td>
<td>6</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>US Legislative Data Group</td>
<td>6</td>
</tr>
<tr>
<td>40</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>10</td>
<td>9</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>US Legislative Data Group</td>
<td>9</td>
</tr>
<tr>
<td>50</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>15</td>
<td>11</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>US Legislative Data Group</td>
<td>11</td>
</tr>
<tr>
<td>60</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>20</td>
<td>16</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>US Legislative Data Group</td>
<td>16</td>
</tr>
<tr>
<td>70</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>69</td>
<td>21</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>US Legislative Data Group</td>
<td>21</td>
</tr>
<tr>
<td>80</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1000</td>
<td>1</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>1</td>
</tr>
<tr>
<td>90</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>2000</td>
<td>4</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>4</td>
</tr>
<tr>
<td>100</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>3000</td>
<td>6</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>6</td>
</tr>
<tr>
<td>110</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>4000</td>
<td>9</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>9</td>
</tr>
<tr>
<td>120</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>5000</td>
<td>11</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>11</td>
</tr>
<tr>
<td>130</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>6000</td>
<td>16</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>16</td>
</tr>
<tr>
<td>140</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>10000</td>
<td>21</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>21</td>
</tr>
<tr>
<td>150</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>500</td>
<td>1</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>1</td>
</tr>
<tr>
<td>160</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1000</td>
<td>4</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>4</td>
</tr>
<tr>
<td>170</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1500</td>
<td>6</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>6</td>
</tr>
<tr>
<td>180</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>2000</td>
<td>9</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>9</td>
</tr>
<tr>
<td>190</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>2500</td>
<td>11</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>11</td>
</tr>
<tr>
<td>200</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>750</td>
<td>4</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>TECHNICAL</td>
<td>1</td>
</tr>
<tr>
<td>210</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>250</td>
<td>6</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>TECHNICAL</td>
<td>4</td>
</tr>
<tr>
<td>220</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>750</td>
<td>9</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>TECHNICAL</td>
<td>6</td>
</tr>
<tr>
<td>230</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1250</td>
<td>11</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>TECHNICAL</td>
<td>9</td>
</tr>
<tr>
<td>240</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1500</td>
<td>16</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>16</td>
</tr>
<tr>
<td>250</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>2500</td>
<td>21</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>EXECUTIVE</td>
<td>21</td>
</tr>
<tr>
<td>260</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>500</td>
<td>1</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>1</td>
</tr>
<tr>
<td>270</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1250</td>
<td>4</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>4</td>
</tr>
<tr>
<td>280</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>1500</td>
<td>6</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>6</td>
</tr>
<tr>
<td>290</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>2500</td>
<td>9</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>9</td>
</tr>
<tr>
<td>300</td>
<td>4712/12/31</td>
<td>2019/06/14</td>
<td>2500</td>
<td>11</td>
<td>VISION_RANGE_OF_ROW_VALUES</td>
<td>MANAGER</td>
<td>11</td>
</tr>
</tbody>
</table>

**Related Topics**
- User Table Validation Formula Type
Example of Loading User-Defined Tables for Matched Row Values

Vision Corporation wants to use a user-defined table to define schedules for their employees. All workers in the company work on a 10 hour a day, four day in a week rotating schedule. The company has these four schedules.

- Monday - Thursday
- Tuesday - Friday
- Wednesday - Saturday
- Thursday - Sunday

As this image shows, this user-defined table contains the schedules available in the organization. The rows contain the name of a day of the week and the columns contain the schedule. The values represent the number of hours to work each day in a schedule.

Use this .dat file to create a user-defined table to store values for worker’s schedules.

```
METADATA|UserDefinedTable|UserTableCode|RangeOrMatch|UserKeyUnits|UserRowTitle|UserTableName|
LegislativeDataGroupName
METADATA|UserDefinedTableColumn|UserColumnCode|DataType|UserColumnName|UserTableCode|LegislativeDataGroupName|
FormulaName
METADATA|UserDefinedTableRow|DisplaySequence|EffectiveEndDate|EffectiveStartDate|RowHighRange|RowLowRangeOrName|
UserTableCode|LegislativeDataGroupName|RowName
METADATA|UserDefinedTableColumnInstance|EffectiveEndDate|EffectiveStartDate|Value|LegislativeDataGroupName|
UserColumnCode|RowLowRangeOrName|UserTableCode
```
Loading Payroll Setup Objects
Related Topics

• User Table Validation Formula Type

Fast Formulas

Example of Loading Fast Formula

Use HCM Data Loader to create formula that are generic expressions of calculations or comparisons that you want to repeat with different input variables. You can use these predefined rules to perform calculations, such as a legislative specific tax or social insurance calculations. Create formula with effective start dates on or before the start dates of other objects that refer to your formula.

Supply the fast formula text in a separate text file and then pass the text file name to the Comments attribute. Further, place the formula text file in the ClobFiles folder within the same compressed file as FastFormula.dat file that references it. Also, don’t supply the text directly in the FastFormula.dat file.

This example loads the fast formula.

```
METADATA|FastFormula|SourceSystemOwner|SourceSystemId|EffectiveStartDate|FormulaCode|FormulaTypeCode|
|FormulaName|FormulaText|LegislativeDataGroupId(SourceSystemId)
MERGE|FastFormula|VISION|MGR_SCHED_HRS|2000/01/01|MGR_RANGE_SCHD_HRS|Range of Scheduled Hours|Manager Range of Scheduled Hrs.txt|
```

Related Topics

• Overview of Using Formulas
• Overview of Writing Formulas

Example of Loading Fast Formula Globals

Use fast formula globals in multiple formulas to support the Human Capital Management business rules. For example, the global value could hold tax relief percentages, bonus percentage or overtime rates. Create globals with effective start dates on or before the formula that refer to them.

This example loads global values using source keys.

```
```
Create the legislative data group using the example provided in the Loading Legislative Data Groups: Examples topic.

**Related Topics**

- Overview of Using Formulas
- Overview of Writing Formulas

### Example of Loading Payroll Time Cards

Use HCM Data Loader to import time cards from a third-party time collection device to the Payroll cloud. Let’s look at this example. Recently, Vision Corp has acquired a company and imports the acquired employees’ approved time to Payroll for payments.

This `PayrollTimeCard.dat` file uses source keys to import approved time entries into the Oracle Payroll Cloud.

```plaintext
METADATA|PayrollTimeCard|SourceSystemOwner|SourceSystemId|LegislativeDataGroupName|AssignmentId(SourceSystemId)|EffectiveStartDate|EffectiveEndDate|TimeCardId
MERGE|PayrollTimeCard|VISION|9999890|Vision Corporation US LDG|HDL034233|2020/01/11|2020/01/18|6
METADATA|TimeEntry|SourceSystemOwner|SourceSystemId|LegislativeDataGroupName|AssignmentId(SourceSystemId)|EffectiveStartDate|EffectiveEndDate|TimeCardInstanceId(SourceSystemId)|TimeType|Time|UnitOfMeasure|TimeCardId|TimeEntryId|Periodicity|Factor|RateDefinitionId|RateValue
MERGE|TimeEntry|VISION|8888812|Vision Corporation US LDG|HDL034233|2020/01/11|2020/01/11|9999890|NB_ZHRX_OTL_CDRM|8.0|H_DECIMAL3|6|6|HOURLY|1.0|300100122876642|7.0
MERGE|TimeEntry|VISION|8888813|Vision Corporation US LDG|HDL034233|2020/01/12|2020/01/12|9999890|NB_ZHRX_OTL_CDRM|8.0|H_DECIMAL3|6|7|HOURLY|1.0|300100122876642|7.0
MERGE|TimeEntry|VISION|8888814|Vision Corporation US LDG|HDL034233|2020/01/13|2020/01/13|9999890|NB_ZHRX_OTL_CDRM|8.0|H_DECIMAL3|6|8|HOURLY|1.0|300100122876642|7.0
MERGE|TimeEntry|VISION|8888815|Vision Corporation US LDG|HDL034233|2020/01/14|2020/01/14|9999890|NB_ZHTL_CDRM|8.0|H_DECIMAL3|6|9|HOURLY|1.0|300100122876642|7.0
METADATA|TimeEntryProperty|SourceSystemOwner|SourceSystemId|LegislativeDataGroupName|AssignmentId(SourceSystemId)|EffectiveStartDate|EffectiveEndDate|TimeType|PropertyName|PropertyValue|TimeCardId|TimeEntryId
MERGE|TimeEntryProperty|VISION|777712|Vision Corporation US LDG|HDL034233|2020/01/11|2020/01/11|NB_ZHRX_OTL_CDRM|NB_ZHRX_OTL_CDRM_State|12|6|9999890
```

This example uses user keys to reference the time card data.

```plaintext
METADATA|PayrollTimeCard|LegislativeDataGroupName|AssignmentNumber|EffectiveStartDate|EffectiveEndDate|TimeCardId
MERGE|PayrollTimeCard|Vision Corporation US LDG|E15CDRM|2020/01/11|2020/01/18|6
METADATA|TimeEntry|LegislativeDataGroupName|AssignmentNumber|EffectiveStartDate|EffectiveEndDate|TimeType|Time|UnitOfMeasure|TimeCardId|TimeEntryId|Periodicity|Factor|RateName|RateValue
MERGE|TimeEntry|Vision Corporation US LDG|E15CDRM|2020/01/11|2020/01/11|NB_ZHRX_OTL_CDRM|8.0|H_DECIMAL3|6|6|HOURLY|1.0|COST_PROCESS_ELEMENT|7.0
```

This example uses user keys to reference the time card data.
Related Topics

- Importing Time Card Entries to Payroll: Procedure
- Processing Time Entries in Payroll

Guidelines for Loading Calculation Cards

Calculation cards capture values required for payroll calculation of some earnings and deductions, such as absence payments and involuntary deductions. Various types of calculation cards exist. For example, UK calculation cards include those for statutory deductions, automatic enrollment in pensions, court orders and student loans, and benefits and pensions. Most legislations support the Calculation Card object. However, the types of calculation cards that are supported vary by legislation. This topic describes some general considerations that apply to all calculation cards when you load them using HCM Data Loader. For legislation-specific information, see All White Papers for Oracle Fusion Applications HCM (Document ID 1504483.1) on My Oracle Support at https://support.oracle.com.

Each calculation card has one or more components, and each component may have one or more sets of component details, or value definition overrides. For example, in the UK court orders and student loans calculation card, each component corresponds to a different type of court order. You enter information about that court order in the component details and value definition overrides.

Calculation cards are used mainly at the level of the payroll relationship. Depending on the legislation, they can also be used at the level of the payroll statutory unit or tax reporting unit.

Key Support

Calculation Card objects are integration-enabled. They support all key types that HCM Data Loader supports.

Using the Reconcile Integration Keys Process

Source keys are supported for integration-enabled objects only. Any business-object occurrences that were created before the Calculation Card business object was enabled for integration may not have SourceSystemOwner and SourceSystemId values. To supply source keys for these objects, run the Reconcile Integration Keys process for individual components of the Calculation Card business object. This process allocates default source keys to any instance of a component without a source key.

💡 Tip: If required, you can update these default source keys later using the Source Key object.

Run the Reconcile Integration Keys process once only for each component of the Calculation Card object. This table lists the components with their integration object names. You select the integration object name when running the Reconcile Integration Keys process.

<table>
<thead>
<tr>
<th>Calculation Card Component</th>
<th>Integration Object Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculation Card</td>
<td>CalculationCard</td>
</tr>
<tr>
<td>Card Association</td>
<td>CalculationCardAssociations</td>
</tr>
<tr>
<td>Component Association</td>
<td></td>
</tr>
<tr>
<td>Card Association Detail</td>
<td>CalculationCardAssociationDetails</td>
</tr>
</tbody>
</table>
### Changing Logical Start and End Dates

In general, you can change the first **effective start date** and last **effective end date** of an existing calculation card. Include the ReplaceFirstEffectiveStartDate and ReplaceLastEffectiveEndDatet attributes in your METADATA line, as appropriate, and set them to `Y`. Provide the new values on the EffectiveStartDate and EffectiveEndDate attributes.

🔥 Note: Legislation-specific validations may mean that some changes to logical start and end dates aren’t valid.

### Deleting Calculation Cards

You can delete all components of the Calculation Card object using HCM Data Loader. When you delete a Calculation Card component, its child components are deleted automatically. You can also delete individual child components.

**Related Topics**

- HCM Data Loader: Loading Calculation Cards for Australia (2124649.1)
- HCM Data Loader: Loading Calculation Cards for France (2233630.1)
- HCM Data Loader: Loading Calculation Cards for United Kingdom (2098480.1)

### Example of Loading Wage Basis Rules

Use HCM Data Loader to load wage basis rules to determine the earnings that are subjected to a deduction. To illustrate how wage basis rules affect a tax calculation, let’s look at an example where an employee’s earnings included in the wage basis vary, depending upon where the employee lives.

Brittany is a salesperson who receives a salary of 2,000 each month. Brittany also has a company car she drives. She is responsible for reporting her personal use of the company car in order to be taxed correctly.

Brittany works in her company’s East Coast district. She spends 50 percent of her time in New York and 50 percent of her time in Pennsylvania. The State of New York taxes Brittany for her personal use of the company car; however, Pennsylvania does not.

Last month, Brittany reported personal use that equated to 100 (50 personal use in New York and 50 personal use in Pennsylvania).

In New York, the imputed earnings, such as personal use of company car amounts are included in the taxable wages. Whereas, in Pennsylvania, these earnings are not included in taxable wages.
This table shows the tax calculations that apply for each region.

<table>
<thead>
<tr>
<th>Region</th>
<th>Earnings in Salary</th>
<th>Eligible Imputed Earnings</th>
<th>Taxable Income</th>
<th>Deduction Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>New York</td>
<td>1000</td>
<td>50</td>
<td>1050</td>
<td>35</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>1000</td>
<td>50 - Exempt</td>
<td>1000</td>
<td>30</td>
</tr>
</tbody>
</table>

Here are the wage basis rule for this tax calculation.

<table>
<thead>
<tr>
<th>Region (Reference Value)</th>
<th>Primary Classification</th>
<th>Secondary Classification</th>
<th>Use in Wage Basis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>New York</td>
<td>Standard Earnings</td>
<td>Regular</td>
<td>Y</td>
</tr>
<tr>
<td>New York</td>
<td>Imputed Earnings</td>
<td>Personal Use of Company Car</td>
<td>Y</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>Standard Earnings</td>
<td>Regular</td>
<td>Y</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>Imputed Earnings</td>
<td>Personal Use of Company Car</td>
<td>N</td>
</tr>
</tbody>
</table>

Use this TaxabilityRule.dat to load the wage basis rules.

```
METADATA|TaxabilityRule|EffectiveStartDate|LegislativeDataGroupName|PayrollComponent|PrimaryClassification|SecondaryClassification|Context1|UsageType
MERGE|TaxabilityRule|2011/1/10|Vision Corp|State Tax|Standard Earnings|Regular|NY|P
MERGE|TaxabilityRule|2011/1/10|Vision Corp|State Tax|Imputed Earnings|Personal Use of Company Car|NY|P
MERGE|TaxabilityRule|2011/1/10|Vision Corp|State Tax|Standard Earnings|Regular|PA|P
MERGE|TaxabilityRule|2011/1/10|Vision Corp|State Tax|Imputed Earnings|Personal Use of Company Car|PA|N
```

Related Topics

- Wage Basis Rules

## Elements

### Overview

Use HCM Data Loader to load elements, which determine the payment of base pay, benefits, absences, earnings and deductions. For example, you can load earnings and deduction elements, such as bonuses, overtime earnings, and involuntary deductions.

You create elements by using the element templates. The design of the template varies depending upon the primary and secondary classifications and the element category. The answers that you load to the template questions determine the components that are created for the element, such as the balances, balance feeds, and formulas required for processing.

Refer to the Element Creation: Review page for a specific list of questions that you will need to answer to load your elements.
This figure illustrates an example of an earnings element for hourly wages. For each component, you will load the respective value. For example, you load Standard Earnings as the value for the component Classification.

This table summarizes the various building blocks of an element:

<table>
<thead>
<tr>
<th>Component</th>
<th>What It Does</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>Determines the processing order and balance feeds for elements. Elements can belong to three types of classifications - Primary, Secondary, and Subclassifications.</td>
</tr>
<tr>
<td>Input Values</td>
<td>Defines values that hold information to calculate an element's payroll run result.</td>
</tr>
<tr>
<td>Eligibility</td>
<td>Determines rules that indicate which employees are eligible for an element.</td>
</tr>
<tr>
<td>Processing Rules</td>
<td>Identifies the formula to create and then use when processing element entries. You can load multiple processing rules if you want to use different formulas, such as a rule depending on an assignment status.</td>
</tr>
<tr>
<td>Formulas</td>
<td>Specifies the calculation used to process elements. An element has one or more formula result rules automatically generated when you create an element.</td>
</tr>
<tr>
<td>Balances</td>
<td>Accumulates result totals over a period of time.</td>
</tr>
</tbody>
</table>

Related Topics
- Elements
Example of Creating Earnings Element for Payroll

This example shows how to use the element template to create a regular earnings element, such as salary. After you create the earnings element, you must create at least one eligibility record for it.

Load these details:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legislative Data Group</td>
<td>Vision Corp</td>
</tr>
<tr>
<td>Primary Classification</td>
<td>Standard Earnings</td>
</tr>
<tr>
<td>Category</td>
<td>Standard</td>
</tr>
<tr>
<td>Element Name</td>
<td>Base Salary</td>
</tr>
<tr>
<td>Reporting Name</td>
<td>Base Salary</td>
</tr>
<tr>
<td>Effective Date</td>
<td>01/01/2010</td>
</tr>
</tbody>
</table>

The table shows the responses that you provide to these questions:

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>What’s the input currency?</td>
<td>US Dollar</td>
</tr>
<tr>
<td>Should every person eligible for the element automatically receive it?</td>
<td>No</td>
</tr>
<tr>
<td>What’s the earliest entry date for this element?</td>
<td>First Standard Earnings Date</td>
</tr>
<tr>
<td>What’s the latest entry date for this element?</td>
<td>Last Standard Earning Date</td>
</tr>
<tr>
<td>At which employment level should this element be attached?</td>
<td>Assignment Level</td>
</tr>
<tr>
<td>Do you want the element to be processed at Payroll Assignment level?</td>
<td>Yes</td>
</tr>
<tr>
<td>Does the element recur each payroll period, or does it require explicit entry?</td>
<td>Recurring</td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>Process the element only once in each payroll period?</td>
<td>Yes</td>
</tr>
<tr>
<td>Can a person have more than one entry of the element in a payroll period?</td>
<td>No</td>
</tr>
<tr>
<td>Process and pay element separately or with other earnings elements?</td>
<td>Process and pay with other earnings</td>
</tr>
<tr>
<td>Tax this earning across multiple pay periods?</td>
<td>No</td>
</tr>
<tr>
<td>Prorate this earning across all periods during which it was earned, and consider it for Overtime calculations, such as for commissions, bonuses, incentives, and other nondiscretionary earnings?</td>
<td>No</td>
</tr>
<tr>
<td>Does this element have a limit on the amount which is exempt from Federal Tax?</td>
<td>No</td>
</tr>
<tr>
<td>Does this element have a limit on the amount which is exempt at state level?</td>
<td>No</td>
</tr>
<tr>
<td>Do you want to default it to Federal level?</td>
<td>Yes</td>
</tr>
<tr>
<td>What’s the calculation rule?</td>
<td>Flat Amount</td>
</tr>
<tr>
<td>What’s the default periodicity of the element?</td>
<td>Periodically</td>
</tr>
<tr>
<td>What’s the periodicity conversion rule?</td>
<td>Standard Rate Annualized</td>
</tr>
<tr>
<td>How do you want the work units to be reported?</td>
<td>Hours</td>
</tr>
<tr>
<td>What’s the work units conversion rule?</td>
<td>Standard Rate Annualized</td>
</tr>
<tr>
<td>Is this element subject to proration?</td>
<td>Yes</td>
</tr>
<tr>
<td>What’s the proration group?</td>
<td>Entry Changes for Proration</td>
</tr>
<tr>
<td>What’s the proration rate conversion rule?</td>
<td>Periodic Work Schedule Rate Annualized</td>
</tr>
</tbody>
</table>
Question | Answer
--- | ---
What’s the proration units? | Hourly
Is this element subject to retroactive changes? | Yes
What’s the retro group? | Entry Change for Retro
Use this element to calculate a gross amount from a specified net amount? | No
Should this element reduce regular earnings? | No
Should this element be included in the earnings calculation of the FLSA overtime base rate? | Yes
Should this element be included in the hours calculation of the FLSA overtime base rate? | Yes

Use this DAT file to create a regular earnings element.

```
METADATA|PayrollElementDetails|ElementName|ReportingName|Description|LegislativeDataGroupName|
PrimaryClassificationName|SecondaryClassificationName|Category|ElementStartDate
METADATA|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementDetails|Base Salary|Base Salary|Base Salary|Vision Corporation US LDG|Standard Earnings||
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementDetails|Base Salary|Base Salary|Base Salary|Vision Corporation US LDG|Standard Earnings||
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
```
Example of Creating a Loan Deduction Element

In this example, an employee takes a personal loan and the Vision Corp company deducts a stipulated amount from their pay for each pay period.

When there are no sufficient earnings to cover the deduction, then the loan amount goes into the employee’s arrears account.

Using HCM Data Loader, load these details:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legislative Data Group</td>
<td>Vision Corp</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Primary Classification</td>
<td>Voluntary Deduction</td>
</tr>
<tr>
<td>Category</td>
<td>Standard</td>
</tr>
<tr>
<td>Element Name</td>
<td>Personal Loan</td>
</tr>
<tr>
<td>Reporting Name</td>
<td>Personal Loan</td>
</tr>
<tr>
<td>Effective Date</td>
<td>01/01/2010</td>
</tr>
</tbody>
</table>

And this table shows the responses that you provide to these questions:

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>What’s the input currency?</td>
<td>US Dollar</td>
</tr>
<tr>
<td>Should every person eligible for the element automatically receive it?</td>
<td>No</td>
</tr>
<tr>
<td>What’s the earliest entry date for this element?</td>
<td>First Standard Earnings Date</td>
</tr>
<tr>
<td>What’s the latest entry date for this element?</td>
<td>Last Standard Process Date</td>
</tr>
<tr>
<td>At which employment level should this element be attached?</td>
<td>Payroll relationship Level</td>
</tr>
<tr>
<td>Do you want the element to be processed at Payroll Relationship level?</td>
<td>Yes</td>
</tr>
<tr>
<td>What should happen when there are insufficient funds to cover the deductions?</td>
<td>Take a partial deduction, place remaining in arrears</td>
</tr>
<tr>
<td>Does the element recur each payroll period, or does it require explicit entry?</td>
<td>Recurring</td>
</tr>
<tr>
<td>Process the element only once in each payroll period?</td>
<td>Yes</td>
</tr>
<tr>
<td>Can a person have more than one entry of the element in a payroll period?</td>
<td>No</td>
</tr>
<tr>
<td>Do you want the element to be processed at Payroll Relationship level?</td>
<td>Yes</td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>What's the calculation rule?</td>
<td>Flat amount deduction</td>
</tr>
<tr>
<td>Is this element subject to proration?</td>
<td>Yes</td>
</tr>
<tr>
<td>Proration Group</td>
<td>Entry Changes for Proration</td>
</tr>
<tr>
<td>Is this element subject to retroactive changes?</td>
<td>Yes</td>
</tr>
<tr>
<td>Retro Group</td>
<td>Entry Changes for Retro</td>
</tr>
<tr>
<td>Processing Stop when the Total is reached?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Use this DAT file to create a loan deduction element.

```
METADATA|PayrollElementDetails|ElementName|ReportingName|Description|LegislativeDataGroupName|
PrimaryClassificationName|SecondaryClassificationName|Category|ElementStartDate
METADATA|PayrollElementQuestionnaire|ElementName|LegislativeDataGroupName|RuleCode|Rule|Response
MERGE|PayrollElementDetails|Personal Loan|Personal Loan|Personal Loan|Vision Corporation US LDG|Voluntary Deductions||Standard|2010/01/01
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|SpecCurrency|Input Currency|US Dollar
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Availability Rule|Should every person eligible for the element automatically receive it?|No
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Hire Process|What is the earliest entry date for this element?|First Standard Earning Date
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Terminate Process|What is the latest entry date for this element?|Last Standard Process Date
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Processing Level|At which employment level should this element be attached?|Payroll relationship level
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Processing Rel Level|Do you want the element to be processed at Payroll Relationship level?|Yes
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Arrear Partial Dedn|What should happen when there are insufficient funds to cover the deductions?|Take a partial deduction, place remaining in arrears
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Recurring Or NonRecurring|Does this element recur each payroll period, or does it require explicit entry?|Recurring
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Once Per Period|Process the element only once in each payroll period?|Yes
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Multiple Entries|Can a person have more than one entry of this element in a payroll period?|No
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Processing Rel Level Excl|Do you want the element to be processed at Payroll Relationship level? Exclude|Yes
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|CalculationRuleDedn|What is the calculation rule?|Fixed amount deduction
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Proration|Is this element subject to proration?|Yes
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Proration Formula|Proration Formula
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Proration Group|Proration Group
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Entry Changes for Proration
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Retro|Is this element subject to retroactive changes?|Yes
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|RetroGroup|Retro Group
MERGE|PayrollElementQuestionnaire|Personal Loan|Vision Corporation US LDG|Total Reached|Processing Stop when the Total is reached?|Yes
```
Example of Loading Payroll Element Definitions

Use HCM Data Loader to update or delete existing element definitions. Element definitions include definitions for payroll, human resources, and benefits elements, such as earnings, deductions, and taxation.

You can’t create element definition objects using HCM Data Loader. You can edit the element definition to define the order in which the elements are processed in the payroll run. These rules determine whether the elements can be processed more than once in a period.

Let’s consider the scenario where you define a new input value to an existing voluntary deduction element that doesn’t have an element eligibility. You then create an eligibility, eligibility input values, and sub classification rules as shown in this figure.

These DAT lines update the existing element description and reporting name.

```sql
MERGE PayrollElementDefinition|Vision Loan Element|2010/01/01|| Vision USA LDG|Car Loan Element|Process car loan deduction
```

ORACLE
Input Value
An element’s input values define the entry values available on each entry of this element. Each input value has a unit of measure, such as money or date. Input values can include validations and conditions to control the data entry of the element entry assigned to a person.

These DAT lines create an input value Loan Reference Number for the element Vision Loan Element.

```
METADATA|InputValue|EffectiveStartDate|EffectiveEndDate|InputValueCode|ValueRequiredFlag|
CreateDatabaseItemFlag|UOM|DisplayFlag|AllowUserEntryFlag|SourceSystemOwner|SourceSystemId|ElementCode|
DisplaySequence|LegislativeDataGroupName|Name
MERGE|InputValue|2010/01/01||Loan Reference Number|Y|Y|C|Y|Y|||Vision Loan Element|100| Vision USA LDG|Loan Reference Number
```

Element Eligibility
Element eligibility determines which people are eligible for an element. To determine eligibility, you select the criteria that people must have to receive entries of the element.

These DAT lines create the element eligibility Vision Loan Element Eligibility with the criteria Payroll = Vision Monthly Payroll.

```
METADATA|ElementEligibility|EffectiveStartDate|EffectiveEndDate|ElementEligibilityName|AutomaticEntryFlag|
ElementCode|LegislativeDataGroupName|PayrollCode
MERGE|ElementEligibility|2010/01/01||HDL Loan Element Eligibility|N|HDL Loan Element|PM US Sun Power|HDL Monthly Payroll
```

Formulas
Formulas specify the calculation used to process elements. Define a fast formula Vision Loan Element Formula that calculates the amount using various criteria. It returns the amount to a target element that’s used to deduct the amount for an employee.

You associate the fast formula in the status processing rules and the deduction amount to the target element in formula result rules.

Status Processing Rule
Define rules for processing elements according to specific assignment statuses. For each assignment status, you can specify a different formula to be run for the same element.

These lines creates a status processing rule.

```
METADATA|StatusProcessingRule|EffectiveStartDate|EffectiveEndDate|ElementCode|LegislativeDataGroupName|
AssignmentStatusCode|FormulaCode|BalanceAdjustment
MERGE|StatusProcessingRule|2010/01/01||HDL Loan Element|PM US Sun Power|Inactive - No Payroll|HDL LOAN ELEMENT FORMULA|No
```

Formula Result Rule
A formula result rule defines the processing rules for each element. Formulas are attached to an element for processing according to specific assignment statuses. For each assignment status, it’s possible to specify a different formula to be fired for the same element. For example, salary is calculated differently when the employee is on leave of absence.

These lines create a formula result rule.

```
METADATA|FormulaResultRule|EffectiveStartDate|EffectiveEndDate|ElementCode|LegislativeDataGroupName|
AssignmentStatusCode|ResultReturned|ResultRule|TargetElementCode|TargetInputValueCode|BalanceAdjustment
MERGE|FormulaResultRule|2010/01/01||HDL Loan Element|PM US Sun Power|Inactive - No Payroll|L_VALUE|Indirect Result|SC_HDL_EARN|Pay Value|No
```
Sub Classification Rule
Define sub classification rules for the elements to control the balance it feeds. An element can have multiple subclassifications.

These lines create a sub classification rule **Disposable Income Garnishment** for the **Standard Earnings** element.

```
METADATA|SubClassificationRule|EffectiveStartDate|EffectiveEndDate|ElementCode|LegislativeDataGroupName | ClassificationCode
MERGE|SubClassificationRule|2010/01/01|| Disposable Income Garnishment | Vision USA LDG|Standard Earnings
```

**Related Topics**
- Elements
- How Element Classification Components Work Together
- Formula Result Rules for Elements
- Element Input Values

Example of Loading Payroll Element Run Usage
Use the payroll element run usage object to identify how an element is used with the run type. Run types control the elements and payment types to process in a payroll run.

You can’t change either the first effective start date or last effective end date for an existing payroll element run usage object. Create these objects with effective start dates on or before the start dates of other objects that refer to your payroll element run usage.

Vision Corp processes and pays the bonus amounts separately from the regular earnings. The organization pays regular payroll by EFT and issues check bonuses once a year. For this requirement, they create a separate payment for the bonus element entry, which is marked to pay separately. Using HCM Data Loader, the Payroll Manager creates a bonus element, Annual Bonus, with Supplemental Earnings classification. She supplies these details.

<table>
<thead>
<tr>
<th>Element Detail</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Classification</td>
<td>Supplemental Earnings</td>
</tr>
<tr>
<td>Secondary Classification</td>
<td>Bonus</td>
</tr>
<tr>
<td>Category</td>
<td>Standard</td>
</tr>
<tr>
<td>Process and pay element separately or with other earnings elements?</td>
<td>Process separately and pay separately</td>
</tr>
</tbody>
</table>

Further, she loads the run type usage as **Separate Payment** with Element Usage as **Trigger**. The application processes this element as well as makes payments separately.

This example dat file loads the payroll element run usage object:

```
METADATA|PayrollElementRunTypeUsage|EffectiveStartDate|UsageType|ElementTypeCode|RunTypeCode|LegislativeDataGroupName
```
Element Entries

Overview of Loading and Rolling Back Element Entries

You use element entries to capture earnings, deductions, absences, and basic benefit details for an employee assignment. For example, you can create element entries for an employee's overtime hours or medical premium deduction amount.

The element entry values hold the necessary values for the element type. For example, a salary entry can contain the salary value and the salary frequency.

Using HCM Data Loader, you can load element entries and element entry values.

Element Entry Rollback

You can roll back changes to those element entries that you loaded using the Initiate Data Loader flow. You use the rollback feature of the flow to roll back the newly created data set.

We recommend that you use payroll flows to submit element entry batches if you use element entry events for reporting, proration, or retropay.

You can only roll back new element entries where no changes have been made to it since it was created by HCM Data Loader. If HCM Data Loader application identifies that the element entry has been updated, the rollback fails. The application will not attempt to roll back updates to an existing element entry. Also, it will not attempt to roll back the deletion of an existing element entry.
This figure illustrates how you can roll back element entries depending upon whether the entries were loaded using payroll flows or not.

If you submit an element entry using the **Initiate Data Loader** flow, then you can roll it back either using flows or the web service. Alternatively, if you didn’t submit your element entry using a flow pattern, then you can use only the HCM Data Loader web service to roll it back.

**Rollback and Events**

When an element entry is rolled back, the application also rolls back any unprocessed events and notifications. Consider these scenarios that explain the impact of a roll back on events.

- If the event has an unprocessed status, then the application removes the event. If the event is already processed, then the application lets it stay.
- If you created an element entry, then application removes the event. The application doesn’t remove other types of element entry events, such as an update element entry event.
- If you updated an element entry, then the application doesn’t delete that event.

**Related Topics**

- [Enable Automatic, Multiple, or Additional Element Entries Options](#)
- [Options to Determine an Element’s Latest Entry Date](#)
- [Default Values for Element Entries](#)

**How You Roll Back Element Entries Using Payroll Flows**

If you used payroll flows to load an element entry, then you can roll it back using the flow’s rollback feature. Use the **Initiate Data Loader** flow pattern to roll back element entries that you uploaded using the flow pattern.
These steps describe how you roll back element entries using the payroll flow.

1. As the first step, you create the ElementEntry.dat file to load your element entries and its values. Also, you must compress it to a file name of your choice.

2. Using the File Import and Export task, you upload the dat file to the Oracle Web Center Content Server. Enter the account as hcm/dataloader/import. You must make a note of the Content ID.

3. From the Checklist or Data Exchange work area, you submit the Initiate Data Loader flow to load and submit the element entries data set. When submitting the flow, provide the Content ID from the previous step.

4. To roll back your element entry data, on the Payroll Flow page, select the Initiate Data Loader flow. Select Mark as Complete from the Actions menu. In the Warning dialog box, click Continue. The application rolls back the newly created element entries.

Related Topics

- Enable Automatic, Multiple, or Additional Element Entries Options
- Default Values for Element Entries

Example of Loading Element Entries

Use HCM Data Loader to create element entries. You could also delete element entry data. As deleted records can't be recovered, you must take caution when deleting records. You can't delete and create the same element entry in the same file.

This dat file creates a recurring element entry.

METADATA|ElementEntry|SourceSystemOwner|SourceSystemId|AssignmentNumber|EffectiveStartDate|EffectiveEndDate|ElementName|LegislativeDataGroupName|MultipleEntryCount|AssignmentNumber|InputValueName|ScreenEntryValue

MERGE|ElementEntry|VISION|1008_MISC_1|E1008|2015/01/01|4712/12/31|Misc Allowance|USA LDG|1|E|H

METADATA|ElementEntryValue|SourceSystemOwner|SourceSystemId|ElementEntryId(SourceSystemId)|EffectiveStartDate|EffectiveEndDate|ElementName|LegislativeDataGroupName|MultipleEntryCount|AssignmentNumber|InputValueName|ScreenEntryValue
End Date Recurring Element Entries

To update the last effective end date, include the ReplaceLastEffectiveEndDate attribute in your METADATA line. Provide a Y value to indicate that the EffectiveEndDate supplied is a change to the existing logical end date of your record. You must supply a record for the element entry and every element entry value.

End Date a Recurring Element Entry

This dat file ends an existing open-ended recurring element entry.

Delete an Element Entry

This dat file deletes an existing element entry:

Note: The complete element entry is deleted. You can’t delete individual date-effective updates to an element entry

Related Topics

- Enable Automatic, Multiple, or Additional Element Entries Options
- Options to Determine an Element’s Latest Entry Date
15 Loading Balance Definitions

Examples of Loading Payroll Balance Definitions

A balance definition includes a combination of user-defined naming criteria, one or more balance dimensions, and balance feeds. Most of the balances you require are predefined. Depending on your country extension, use HCM Data Loader to define additional balances for calculations or reporting.

Let’s consider this example. The members of Vision Corps union contribute their union dues that’s determined by a formula. The formula computes the union dues as a percentage of a user-defined Union Dues Basis balance, which sums up these earnings.

- All earnings in the Standard Earnings classification.
- All earnings in the Imputed Earnings classification.
- A specific car allowance element.

This figure includes the balance dimensions that identify the specific value of a balance. Here, the balance is fed by the element classifications and an element.

Payroll balance definitions are made up of these components:

- Balances
Balances

Payroll balances show the accumulation of values over a period of time. Typically, the global and country-specific rules create a majority of balances whenever you create an earnings or deductions element. However, you might need to define a balance definition outside of the element setup.

You supply balance definitions in the PayrollBalanceDefinition.dat file. Here, you assign a category of **Miscellaneous** so that the balance isn’t included in the reports. Since the monetary value is fed into the balance, you supply UOM as **Money** and the Currency as **USD**.

```
MERGE|PayrollBalanceDefinition|SourceSystemOwner|SourceSystemId|LegislativeDataGroupName|BalanceCode|
BalanceName|ReportingName|BalanceCategoryCode|BalanceUOM|CurrencyCode
VISION|VISION|UNION_DUES|Vision Corporation US LDG|HDL_UNION_DUES_BASIS|Union Dues Basis|Union Dues Basis|Miscellaneous|M|USD
```

**Defined Balances**

Each payroll balance can have multiple dimensions, which define the specific value to retrieve. The following Defined Balance lines load these relationship level dimensions for which the balance value is aggregated.

- Relationship Period to Date
- Relationship Month to Date
- Relationship Quarter to Date
- Relationship Year to Date

```
METADATA|DefinedBalance|SourceSystemOwner|SourceSystemId|BalanceId(SourceSystemId)|LegislativeDataGroupName|BalanceDimensionCode
VISION|VISION|UNION_DUES_DB_PTD|UNION_DUES|Vision Corporation US LDG|Relationship Period to Date
VISION|VISION|UNION_DUES_DB_MTD|UNION_DUES|Vision Corporation US LDG|Relationship Month to Date
VISION|VISION|UNION_DUES_DB_QTD|UNION_DUES|Vision Corporation US LDG|Relationship Quarter Year to Date
VISION|VISION|UNION_DUES_DB_YTD|UNION_DUES|Vision Corporation US LDG|Relationship Year to Date
VISION|VISION|UNION_DUES_DB_RUN|UNION_DUES|Vision Corporation US LDG|Relationship Run
```

**Balance Feed**

A balance feed contains details of how a given element input value contributes to a specific balance. You can load element classifications and individual elements to feed a balance.

You can’t change either the first effective start date or last effective end date for an existing balance feed. Create balance feed objects with effective start dates on or before the start dates of other objects that refer to your balance feed.

**Balance Feed by Classification**

These balance classification lines create balance classifications, which determine the element classifications you add to or subtract from the balance. The direct run result value of every element in the classification feeds the balance. Here, you load **Standard Earnings** and **Imputed Earnings** element classifications with Add feed.

```
METADATA|BalanceClassification|SourceSystemOwner|SourceSystemId|BalanceId(SourceSystemId)|LegislativeDataGroupName|ElementClassificationCode|AddSubtract
VISION|VISION|UNION_DUES_BC_STD|UNION_DUES|Vision Corporation US LDG|Standard Earnings|1
VISION|VISION|UNION_DUES_BC_TAX|UNION_DUES|Vision Corporation US LDG|Taxable Benefits|1
```
Balance Feed by Elements

Load individual elements and input values to feed the balance.

These balance feed lines add an element feed from the **ZFRC VS USD Car Allowance Element Earnings Results** element and **Earnings Calculated** input value.

```plaintext
METADATA|BalanceFeed|SourceSystemOwner|SourceSystemId|BalanceId(SourceSystemId)|EffectiveStartDate|
LegislativeDataGroupName|ElementCode|InputValueCode|AddSubtract
MERGE|BalanceFeed|VISION|UNION_DUES_BF_IMP|UNION_DUES|2018/01/01|Vision Corporation US LDG|ZFRC VS USD Car Allowance Element Earnings Results|Earnings Calculated|1
```

Related Topics

- Payroll Balance Definitions
- Balance Feeds

Payroll Balance Groups

Example of Loading Payroll Balance Attribute Definitions

Use HCM Data Loader to define balance attribute definitions and add balance definitions to it.

A balance attribute definition is a set of balance definitions that you can include in your balance group. It identifies the criteria for including the balance definitions. Use a balance attribute to associate a balance definition to a balance attribute definition.

In this example, the Vision Corp Company wants to report a particular statutory deduction balance for employees. To do this, first you create a balance attribute definition **Employee Taxes Attribute Definition** and add these balance definitions to it.

<table>
<thead>
<tr>
<th>Balance Type</th>
<th>Balance Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIT Withheld</td>
<td>Relationship Tax Unit Period to Date</td>
</tr>
<tr>
<td>Medicare Employee Withheld</td>
<td>Relationship Tax Unit Period to Date</td>
</tr>
<tr>
<td>Social Security Employee Withheld</td>
<td>Relationship Tax Unit Period to Date</td>
</tr>
<tr>
<td>FIT Withheld</td>
<td>Relationship Tax Unit Quarter to Date</td>
</tr>
<tr>
<td>Medicare Employee Withheld</td>
<td>Relationship Tax Unit Quarter to Date</td>
</tr>
<tr>
<td>Social Security Employee Withheld</td>
<td>Relationship Tax Unit Quarter to Date</td>
</tr>
<tr>
<td>FIT Withheld</td>
<td>Relationship Tax Unit Year to Date</td>
</tr>
<tr>
<td>Medicare Employee Withheld</td>
<td>Relationship Tax Unit Year to Date</td>
</tr>
</tbody>
</table>
The next step is to create a balance group and add this balance attribute definition to the balance group. Later you create a report sort type to sort the balance values and then create a balance group usage for your balance group.

Creating a Balance Attribute Definition

In the **BalanceAttributeDefinition.dat** file you define a balance attribute definition and add balance definitions to it.

```
MERGE|BalanceAttributeDefinition|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|Employee Taxes Attribute Definition|Y|Attribute definition for Employee Taxes reporting
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_FIT_WITHHELD|Core Relationship Tax Unit Period to Date
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_FIT_WITHHELD|Core Relationship Tax Unit Quarter Year to Date
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_FIT_WITHHELD|Core Relationship Tax Unit Year to Date
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_MEDICARE_EMPLOYEE_WITHHELD|Core Relationship Tax Unit Period to Date
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_MEDICARE_EMPLOYEE_WITHHELD|Core Relationship Tax Unit Quarter Year to Date
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_MEDICARE_EMPLOYEE_WITHHELD|Core Relationship Tax Unit Year to Date
MERGE|BalanceAttribute|Vision Corporation US LDG|EMP_TAXES_ATTR_DEFN|US_SOCIAL_SECURITY_EMPLOYEE_WITHHELD|Core Relationship Tax Unit Period to Date
```

---

<table>
<thead>
<tr>
<th>Balance Type</th>
<th>Balance Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Security Employee Withheld</td>
<td>Relationship Tax Unit Year to Date</td>
</tr>
</tbody>
</table>

---
Example of Loading Payroll Balance Groups

Use HCM Data Loader to define payroll balance groups for viewing and reporting payroll balances. You can define a balance group either at the employee level or at group level.

In this example, you define a payroll balance group Employee Taxes Balance Group and add the balance attribute definition to it.

Before you do this, ensure that you have created the balance attribute definition Employee Taxes Attribute Definition. Also, you must have added balances to the attribute definition.

Refer to the topic: Loading Payroll Balance Attribute Definitions for more information on creating a balance attribute definition.
After defining a payroll balance group, the next step is to create a sort list to sort the balance items. Then you create a balance group usage and add this balance group to a balance group usage.

Creating a Payroll Balance Group

Use the **BalanceGroup.dat** file to create a payroll balance group and include the balance attribute definition in it.

```
METADATA|BalanceGroup|LegislativeDataGroupName|BalanceGroupCode|GroupName|Description|
BalanceCategoryRestricted|BalanceDimensionRestricted
MERGE|BalanceGroup|Vision Corporation US LDG|EMP_TAXES_BAL_GRP|Employee Taxes Balance Group|Balance group for Employee Taxes reporting|N|N
METADATA|BalanceGroupInclusion|LegislativeDataGroupName|BalanceGroupCode|AttributeCode
MERGE|BalanceGroupInclusion|Vision Corporation US LDG|EMP_TAXES_BAL_GRP|EMP_TAXES_ATTR_DEFN
```

**Related Topics**

- Balance Groups and Usages
- Create Balance Groups and Usages
- Rules for Editing Balance Groups and Their Usages

Example of Loading Report Sort Type and Report Sort Items

Use HCM Data Loader to create a report sort type and items that defines how balance values are sorted in the report.

Let’s look at this example. Vision Corp creates a report sort type and specifies the sort type as Static Order to define the sequence that displays the balance results.

**Before You Start**

Ensure that you have completed these tasks:

- Created the balance attribute definition **Employee Taxes Attribute Definition**.

  For more information on how to create balance attribute definitions, refer to the topic: Loading Balance Attribute Definitions.

- Created the balance group **Employee Taxes Balance Group** and included the balance attribute definitions to it.
For more information on how to create and load a balance group using HCM Data Loader, refer to the topic: Loading Payroll Balance Groups.

After creating a report sort type, you create a balance group usage for your balance group and define the balance group usage items. For more information on how to create a balance group usage using HCM Data Loader, refer to the topic: Loading Payroll Balance Groups Usage.

Creating a Report Sort Type

Use the `ReportSort.dat` file to create a report sort type with sort method as Static Order.

```
METADATA|ReportSortType|LegislativeDataGroupName|ReportSortCode|SortName|Description|SortMethod|SortLevel|SortOrder
MERGE|ReportSortType|Vision Corporation US LDG|EMP_TAXES_STATIC_SORT|Employee Taxes Sort|Static sort order for Employee Taxes|Static Order|Balance Type|Ascending
```

Creating Report Sort Items

Use the `ReportSortItem.dat` file to specify the order in which balance types are displayed for the balance group usage.

```
METADATA|ReportSortItem|LegislativeDataGroupName|ReportSortCode|SequenceNumber|BalanceCode
MERGE|ReportSortItem|Vision Corporation US LDG|EMP_TAXES_STATIC_SORT|10|US_FIT_WITHHELD
MERGE|ReportSortItem|Vision Corporation US LDG|EMP_TAXES_STATIC_SORT|20|US_MEDICARE_EMPLOYEE_WITHHELD
MERGE|ReportSortItem|Vision Corporation US LDG|EMP_TAXES_STATIC_SORT|30|US_SOCIAL_SECURITY_EMPLOYEE_WITHHELD
```

Related Topics

- Balance Groups and Usages
• Create Balance Groups and Usages
• Rules for Editing Balance Groups and Their Usages

Example of Loading Payroll Balance Group Usages

Use HCM Data Loader to create balance group usages that represent an instance of a balance group, such as report, extract definition, and so on. For every balance group, you must define at least one balance group usage. But you can define multiple usages for a balance group.

Vision Corp creates a balance group usage to use the **Employee Taxes Balance Group** balance group for their employee taxes report. Include your report sort type **Employee Taxes Sort** to this balance group usage.

**Before You Start**

Ensure that you have completed these tasks:

• Created the balance attribute definition **Employee Taxes Attribute Definition**. For more information on how to create balance attribute definitions, refer to the topic: Loading Balance Attribute Definitions.
• Created the balance group **Employee Taxes Balance Group** and included the balance attribute definitions to it. For more information on how to create and load a balance group using HCM Data Loader, refer to the topic: Loading Payroll Balance Groups.
• Created the Report Sort Type **Employee Taxes Sort** and added the sort items to it. For more information on how to create report sort type, refer to the topic: Loading Report Sort.
Creating a Payroll Balance Group Usage

This example `BalanceGroupUsage.dat` file creates the Employee Period Taxes Balance group Usage in a matrix format. Each balance group usage item defines a column in the Employee Period Taxes balance results matrix.

```
METADATA|BalanceGroupUsage|LegislativeDataGroupName|BalanceGroupUsageCode|GroupUsageName|Description|
FormatType|BalanceGroupCode|ReportSortCode
MERGE|BalanceGroupUsage|Vision Corporation US LDG|EMP_PERIOD_TAXES|Employee Period Taxes|Employee taxes matrix with static sort order|MATRIX|EMP_TAXES_BAL_GRP|EMP_TAXES_STATIC_SORT
METADATA|BalanceGroupUsageItem|LegislativeDataGroupName|BalanceGroupUsageCode|SourceType|Position|
BalanceDimensionCode
MERGE|BalanceGroupUsageItem|Vision Corporation US LDG|EMP_PERIOD_TAXES|Balance Dimension|10|Core
Relationship Tax Unit Period to Date
MERGE|BalanceGroupUsageItem|Vision Corporation US LDG|EMP_PERIOD_TAXES|Balance Dimension|20|Core
Relationship Tax Unit Quarter Year to Date
MERGE|BalanceGroupUsageItem|Vision Corporation US LDG|EMP_PERIOD_TAXES|Balance Dimension|30|Core
Relationship Tax Unit Year to Date
```

Related Topics

- Balance Groups and Usages
- Create Balance Groups and Usages
- Rules for Editing Balance Groups and Their Usages

Balance Initializations

Overview

When migrating payroll data from another application into cloud payroll, you might have to set the initial balance values. To do this, you can use HCM Data Loader to load balance values into batch views.

After loading the balance values, you submit the Load Initial Balances process. The process validates the batch data, and then processes the batch to load the balance data. It then creates balance adjustments to set the required balance values.

For each balance to initialize, the Load Initial Balances process defines these elements automatically with Balance Initialization classification. And adds them to the balance as balance feeds.

> Note: You can’t initialize balances after the payroll is run for the employee. In such a case, you will have to roll back the payroll run or do a balance adjustment.

Example

In the middle of the year, your company hired a group of employees as part of an acquisition. The original company already paid them their earnings and withheld their taxes. In your company’s ongoing payroll, you can initialize these employees’ payroll balances for that year. This way, you ensure that all further statutory reports are accurate for the entire year and not just for the period they worked in your company.
As this image shows, you can initialize the total earnings balances for period-to-date, quarter-to-date, and year-to-date as of 18-Jun.

### Initial balance values in the interface tables

<table>
<thead>
<tr>
<th>1-Jul</th>
<th>18-Jun</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total_Earnings_PTD</strong> = 100</td>
<td></td>
</tr>
<tr>
<td><strong>Total_Earnings_QTD</strong> = 250</td>
<td></td>
</tr>
<tr>
<td><strong>Total_Earnings_YTD</strong> = 500</td>
<td></td>
</tr>
</tbody>
</table>

This image shows the initializing results as of 18-Jun.

### Initialization Results

- **Total_Earnings_YTD** = 250
- **Total_Earnings_QTD** = 150
- **Total_Earnings_PTD** = 100

Use the InitializeBalanceBatchHeader.dat file to create the VisionBatch batch.

```plaintext
METADATA|InitializeBalanceBatchHeader|LegislativeDataGroupName|BatchName|UploadDate
MERGE|InitializeBalanceBatchHeader|Vision Corporation US LDG|VisionBatch|2018/06/18|
```

In the InitializeBalanceBatchLine.dat file you initialize these three balance values for the total earnings balance.

<table>
<thead>
<tr>
<th>Balance</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total_Earnings_PTD</td>
<td>100</td>
</tr>
</tbody>
</table>
Balance | Value
--- | ---
Total_Balance| 250
Total_Balance| 500

When you run the Load Initial Balances process, it sets the values for each balance relative to the upload date. For example, for an upload date of June 18, the balance values include all payment dates up to and including June 18. The process creates date-effective balance entries (adjustments) to ensure the legislative balances are correct.

As this table shows, the balance initialization process creates adjustments on the first day of the time period relevant to each dimension.

<table>
<thead>
<tr>
<th>Adjustment Date</th>
<th>Adjusted Balances</th>
<th>Adjustment Value</th>
</tr>
</thead>
</table>
| 1 June (Start of the pay period) | • Total_Balance_PTD  
• Total_Balance_QTD  
• Total_Balance_YTD | 100 |
| 1 April (Start of the quarter) | • Total_Balance_QTD  
• Total_Balance_YTD | 150 |
| 1 Jan (Start of the year) | Total_Balance_YTD | 250 |

Steps to Initialize Balances

You can use HCM Data Loader to load the initial balance values before you process the employee in a payroll. Before you start initializing the balances, group your employees into batches.
This image shows the steps involved in initializing balances.

1. As the first step, you identify the balances to initialize. You can initialize legislative and user-defined balances. Each of the balances has its own initialization requirements.
2. Map the balances into the Global Payroll cloud. Remember, one legacy balance can feed one or more cloud balances. Also, many legacy balances can feed one cloud balance.
3. Extract legacy balances to load into the payroll cloud.
4. Load balance values to PAY_BAL_BATCH_HEADERS and PAY_BAL_BATCH_LINES interface tables by using these two separate HDL files.
   - InitializeBalanceBatchHeader.dat
   - InitializeBalanceBatchLine.dat
5. Use the Load Initial Balances process from the Payroll Calculation work area to validate and transfer data from the batches into Global Payroll. You can also use the process to roll back the batch if you need to correct and reprocess errors. This process creates date-effective balance entries, or adjustments to ensure that the balances are correct from the upload date.
6. Check the balance load results to identify and correct any errors in the balances. Use the View Person Process Results and View Payroll Process Results tasks from the Payroll Calculation work area to reconcile the loaded balances.

Example of Loading Initial Balances

In this example, you create balance initialization records to initialize an employee's federal and state-level balances. When initializing year-to-date and quarter-to-date balances, you also initialize any related balances. For example, when initializing FIT Withheld balance, related balances such as FIT Exempt and FIT Gross too are initialized.

Load the Balance Batch Header

Use the InitializeBalanceBatchHeader.dat file to create the VisionBatch batch.
Load Batch Lines Data
Load these details to initialize the YTD salary balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>1</td>
</tr>
<tr>
<td>Balance</td>
<td>PM US Sun Power Salary</td>
</tr>
<tr>
<td>Dimension</td>
<td>Assignment Tax Unit Year to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>6000.00</td>
</tr>
</tbody>
</table>

Load these details to initialize the QTD salary balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>2</td>
</tr>
<tr>
<td>Balance</td>
<td>PM US Sun Power Salary</td>
</tr>
<tr>
<td>Dimension</td>
<td>Assignment Tax Unit Quarter to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>6000.00</td>
</tr>
</tbody>
</table>

Load these details to initialize the YTD FIT balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>3</td>
</tr>
<tr>
<td>Balance</td>
<td>FIT Withheld</td>
</tr>
<tr>
<td>Dimension</td>
<td>Relationship Tax Unit Year to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>954.06</td>
</tr>
</tbody>
</table>

Load these details to initialize the QTD FIT balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>4</td>
</tr>
<tr>
<td>Balance</td>
<td>FIT Withheld</td>
</tr>
</tbody>
</table>
Load these details to initialize the YTD Medicare balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>5</td>
</tr>
<tr>
<td>Balance</td>
<td>Medicare Employee Withheld</td>
</tr>
<tr>
<td>Dimension</td>
<td>Relationship Tax Unit Year to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>87.00</td>
</tr>
</tbody>
</table>

Load these details to initialize the QTD Medicare balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>6</td>
</tr>
<tr>
<td>Balance</td>
<td>Medicare Employee Withheld</td>
</tr>
<tr>
<td>Dimension</td>
<td>Relationship Tax Unit Quarter to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>87.00</td>
</tr>
</tbody>
</table>

Load these details to initialize the YTD SIT balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>7</td>
</tr>
<tr>
<td>Balance</td>
<td>SIT Withheld</td>
</tr>
<tr>
<td>Dimension</td>
<td>Relationship Tax Unit,State Year to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>264.00</td>
</tr>
<tr>
<td>Area One</td>
<td>6 (Colorado)</td>
</tr>
</tbody>
</table>
Load these details to initialize the QTD SIT balance.

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>8</td>
</tr>
<tr>
<td>Balance</td>
<td>SIT Withheld</td>
</tr>
<tr>
<td>Dimension</td>
<td>Relationship Tax Unit</td>
</tr>
<tr>
<td></td>
<td>State Year to Date</td>
</tr>
<tr>
<td>Balance Value</td>
<td>264.00</td>
</tr>
<tr>
<td>Area One</td>
<td>6 (Colorado)</td>
</tr>
</tbody>
</table>

In the InitializeBalanceBatchLine.dat file, you initialize the YTD and QTD values for these balances.

- Salary
- FIT
- Medicare
- SIT

METADATA|InitializeBalanceBatchLine|BatchName|LineSequence|PayrollName|PayrollRelationshipNumber|TermNumber|
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AssignmentNumber</td>
<td>BalanceName</td>
<td>DimensionName</td>
<td>TaxUnitName</td>
<td>AreaOne</td>
<td>Value</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>1</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>2</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>3</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>4</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>5</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>6</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>7</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
<tr>
<td>MERGE</td>
<td>InitializeBalanceBatchLine</td>
<td>VisionBatch</td>
<td>8</td>
<td>PM US Sun Power Weekly</td>
<td>955160008191951</td>
</tr>
</tbody>
</table>

Balance Adjustments

Overview

For a given worker, you can use HCM Data Loader to adjust payroll balances as of a certain date. The balance adjustment process creates a run result that adjusts one or more balances depending upon the balance fed by the element input value.

You can decide whether to cost the balance adjustment results or not. Additionally, you can also decide whether you want the adjustment results to be paid or not. You can pay the employee when you run the next pre-payment process that
includes these adjustment.run results. Further, the process assigns the results to the appropriate payment method as per the worker’s payment method setup.

**Corrections to Balance Initialization**

Recollect the scenario where a group of employees were hired through an acquisition in the middle of a year. The original company already paid them their earnings and withheld their taxes. In your company’s ongoing payroll, you initialize these employees' payroll balances for that year. As you can’t initialize balances after the payroll is run for the employee, you adjust the balances for them.

These scenarios describe how you can adjust balances after the first payroll run.

- The FIT Withheld balance for Andrew was initialized 100 less than what it should have been. The Payroll Manager makes a balance adjustment of 100 for Andrew using the payroll element that feeds the FIT Withheld balance.
- Jane Reifer was omitted during the balance initialization process and none of her balances were loaded. The Payroll Manager can adjust Jane’s omitted balances.

**Third-Party Gross to Net calculations**

There are several kinds of earnings that are managed by a third-party. For example, a company could use a third-party to handle disability payments, stock option payouts, moving and relocation expenses. In these cases, the employer still does the reporting of wages or taxes. Also, they adjust the balances by importing the incremental values, which the third-party calculates and withholds, to the application. The company can cost these balance adjustments as the third-party has already made payments to the employee.

Let’s consider this example. Prasad is on a long term disability leave and his employer uses a third-party to manage his disability leave payments.

The third-party does these tasks.

- Calculates Prasad's gross disability earnings as 5000 for the pay period.
- Withholds FIT, SS and Medicare taxes as 1000, 500 and 250, respectively.
- Pays him a net amount of 3250.

The Payroll Manager imports these values as balance adjustments for the balances.

**Retrospective Changes Not Covered by RetroPay**

Typically, you use the retroactive pay process to handle retroactive changes to data that impacts payroll calculation results. However, not all elements support Retro Pay. For example, tax deductions are not enabled for retroactive pay. In such cases, for a retroactive change that impacts tax calculations, you can reflect the changes through balance adjustments.

Let’s consider this example. John lives in California and pays his taxes there. He is on a biweekly payroll and is withheld 500 of the State Income Tax for the two weeks of earnings. In the 16 2018 Biweekly period, he worked in Texas, but didn’t report the time on his time card. He should have been withheld SIT for 200 to Texas instead of the 500 to California.

In John’s process results for the 16th payroll run, the payroll manager adjusts the existing SIT Withheld of 500 to California by doing these actions:

- Make a 200 adjustment to Work State Income Tax for the state of Illinois.
- Negate the original run result for the amount 500.
Zeroing out Balances for Terminations
Here's an example where you zero out the balances for a terminated employee.

James Nance is terminated as of 9/15/2018 with a final close date of 9/30/2018. The payroll manager would like to zero out all arrears balances for James as of 9/30. Upon termination, the manager runs the deductions report and finds that James has these arrears balances as of 9/30.

- Car Loan Deduction 2000.00
- Student Loan Deduction 500.00

For the Arrears Adjustment element, the payroll manager loads -2000.00 and -500 as input values. These values zero out the Inception to Date arrears balances.

Related Topics
- Examples of Balance Exceptions

Steps to Adjust Balances
You can use HCM Data Loader to load balance adjustments in the batch mode.

Before You Begin
Before you adjust balances, do these tasks:

- Create the appropriate eligibility links for the elements that you are adjusting. For example, to adjust base elements, it must contain an eligibility link.
- Populate the Balance Date column if the balance dimension requires an entry.
Perform these steps to adjust your balances.

1. Determine the balances that require an adjustment.
2. Create the batch header and detail lines in the dat files. Use the balance adjustment header to adjust balances in bulk. The balance adjustment lines contain the individual adjustment lines for the balance adjustment group.
3. Import and load the data using the HCM Data Load process. This process creates a run result by adjusting one or more balances. And this adjustment depends upon the balance fed by the element input value.
4. Decide whether the balance adjustment results must be costed or not. If you want to cost your balance adjustments, you can create the cost allocation dat files.
Note: Not all costed adjustments need the cost allocation files. Only those costed adjustments that need one or more costing segments to be overridden for the adjustment element entry need the allocation file.

5. From Payroll Calculation work area, run the Adjust Multiple Balances process to create balance adjustment element entries for each line within the data set. In this process, you provide the batch name that you used in the HCM Data Loader file.

6. Verify the balance results of adjustments.

Related Topics
- Examples of Balance Exceptions

Balance Adjustment Record Types

Use the Balance Adjustment Header and the Balance Adjustment Line business objects to upload the balance adjustment details.
This image shows the hierarchy of components that are applicable to the batch data.

Component: Balance Adjustment Header

- Component: Balance Adjustment Group

Component: Balance Adjustment Line

- Component: Balance Adjustment Value
- Component: Balance Adjustment Value

Component: Balance Adjustment Line

- Component: Balance Adjustment Value

Component: Balance Adjustment Line

- Component: Balance Adjustment Value

Component: Balance Adjustment Line

- Component: Balance Adjustment Value
- Component: Balance Adjustment Value

Component: Balance Adjustment Value repeat as needed

This table summarizes the various balance adjustment components:

<table>
<thead>
<tr>
<th>Component</th>
<th>What It’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balance Adjustment Header</td>
<td>The header for the batch process that is used when adjusting balances in bulk. Supply one batch header record for each group.</td>
</tr>
<tr>
<td>Balance Adjustment Group</td>
<td>A group of balance adjustments within the batch header that correspond to a unique combination of payroll, consolidation group, and effective date. Supply one group record for each batch header. At this time, you may only have one group record per batch.</td>
</tr>
</tbody>
</table>
### Component

<table>
<thead>
<tr>
<th>Component</th>
<th>What It’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balance Adjustment Line</td>
<td>Individual adjustment lines for the balance adjustment group. Each line corresponds to a specific adjustment element entry for a worker and the associated contexts for the adjustment run result.</td>
</tr>
<tr>
<td>Balance Adjustment Value</td>
<td>Individual adjustment values for each balance adjustment line. Each value corresponds to a specific adjustment entry value for a worker and adjusts the balances it feeds.</td>
</tr>
</tbody>
</table>

**Related Topics**
- [Examples of Balance Exceptions](#)

### Example of Loading Balance Adjustments

Vision Corp wants to create balance adjustment records to adjust the FIT Withheld balances for two of their employees. Using HCM Data Loader, they want to increase the withheld balance by 10.

#### Load the Balance Adjustment Header

Use this BalanceAdjustmentHeader.dat file to create the batch header.

```
METADATA|BalanceAdjustmentHeader|LegislativeDataGroupName|BatchName
MERGE|BalanceAdjustmentHeader|US LDG|FIT Adjustments
METADATA|BalanceAdjustmentGroup|LegislativeDataGroupName|BatchName|EffectiveDate|PayrollName|ConsolidationSetName|PrepayFlag|BalanceAdjCostFlag
MERGE|BalanceAdjustmentGroup|US LDG|FIT Adjustments|2019/04/30|Vision US Weekly|Vision Weekly|N|N
```

#### Load the Balance Adjustment Line and Value

For Employee 1 with Assignment number E3263769, load these details:

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>1</td>
</tr>
<tr>
<td>Effective Date</td>
<td>2010/04/30</td>
</tr>
<tr>
<td>Payroll</td>
<td>Vision US Weekly</td>
</tr>
<tr>
<td>Consolidation Group</td>
<td>Vision Weekly</td>
</tr>
<tr>
<td>Assignment Number</td>
<td>E3263769</td>
</tr>
<tr>
<td>Include adjustment in costing process</td>
<td>No</td>
</tr>
<tr>
<td>Federal Income Tax, Tax Calculated</td>
<td>10.00</td>
</tr>
<tr>
<td>Include adjustment in payment balance</td>
<td>No</td>
</tr>
</tbody>
</table>
For Employee 2 with Assignment number E5746169, load these details:

<table>
<thead>
<tr>
<th>Data Column</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Sequence</td>
<td>2</td>
</tr>
<tr>
<td>Effective Date</td>
<td>2010-04-30</td>
</tr>
<tr>
<td>Payroll</td>
<td>Vision US Weekly</td>
</tr>
<tr>
<td>Consolidation Group</td>
<td>Vision Weekly</td>
</tr>
<tr>
<td>Assignment Number</td>
<td>E5746169</td>
</tr>
<tr>
<td>Include adjustment in costing process</td>
<td>No</td>
</tr>
<tr>
<td>Federal Income Tax, Tax Calculated</td>
<td>10.00</td>
</tr>
<tr>
<td>Include adjustment in payment balance</td>
<td>No</td>
</tr>
</tbody>
</table>

Use this BalanceAdjustmentLine.dat file to create the batch lines.

```
METADATA|BalanceAdjustmentLine|LegislativeDataGroupName|BatchName|PayrollName|ConsolidationSetName|EffectiveDate|BatchLineSequence|AssignmentNumber|ElementName|TaxReportingUnitName
METADATA|BalanceAdjustmentValue|LegislativeDataGroupName|BatchName|PayrollName|ConsolidationSetName|EffectiveDate|BatchLineSequence|InputValueName|EntryValue|ElementName
```
16 Loading Payment Methods

Examples of Loading Banks

A bank is a financial institution that can have multiple branches. External bank accounts are associated with bank branches. Bank, Bank Branch, and External Bank Account are business objects whose details are stored in non-HCM tables. However, you can load them using HCM Data Loader. This topic describes aspects of the Bank object that you must understand to load banks successfully.

Loading Banks

You load bank data in a Bank.dat file for processing by HCM Data Loader. To provide a unique reference to a bank when you create it, you must supply either the Bank Name or the Bank Number attribute. You must also supply the Country Code attribute.

This example Bank.dat file creates a Bank object using the Bank Name and Country Code user keys. It also supplies the Bank Number attribute.

```
METADATA|Bank|BankName|BankNumber|CountryCode
MERGE|Bank|Vision Bank|100001A|CA
```

Deleting Banks

You can’t delete Bank objects using HCM Data Loader. However, you can inactivate banks that are either no longer required or were entered in error. This feature, which isn’t available on the Manage Banks pages, enables you to maintain an audit trail.

This example Bank.dat file provides an end date for a Bank object to inactivate it. To identify an existing bank uniquely, you can use the Country Code and either the Bank Name or the Bank Number.

```
METADATA|Bank|BankNumber|CountryCode|EndDate
MERGE|Bank|100001A|CA|2016/03/01
```

Examples of Loading Bank Branches

A bank is a financial institution that can have multiple branches. External bank accounts are associated with bank branches. Bank, Bank Branch, and External Bank Account are business objects whose details are stored in non-HCM tables. However, you can load them using HCM Data Loader. This topic describes aspects of the Bank Branch object that you must understand to load bank branches successfully.

Loading Bank Branches

Bank branch data must be supplied in a BankBranch.dat file for processing by HCM Data Loader. The Bank Branch Number attribute, for which the field name on the Manage Bank Branches page varies by country, uniquely identifies the bank branch. It may be a required attribute, depending on country-specific validations.
You must supply one of the following combinations of attributes to provide a unique reference to the bank branch and the associated bank:

- Bank Branch Name, Country Code, and either Bank Name or Bank Number
- Bank Branch Number, Country Code, and either Bank Name or Bank Number

This example BankBranch.dat file creates a Bank Branch object using the Bank Branch Number, Bank Name, and Country Code user keys.

```
METADATA|BankBranch|BankBranchNumber|BankBranchName|BankName|CountryCode|EftSwiftCode
MERGE|BankBranch|111111A|Toronto 3|Vision Bank|CA|12345678
```

Deleting Bank Branches

You can't delete Bank Branch objects using HCM Data Loader. However, you can inactivate bank branches that are no longer required or were entered in error. This feature, which isn’t available on the Manage Bank Branches pages, enables you to maintain an audit trail.

This example BankBranch.dat file provides an end date to inactivate a bank branch.

```
METADATA|BankBranch|BankBranchNumber|BankName|CountryCode|EndDate
MERGE|BankBranch|111111A|Vision Bank|CA|2015/04/01
```

Examples of Loading External Bank Accounts

An external bank account record holds the details of a bank account at a bank branch. Bank accounts are used by payment methods to make payments to a person. Bank, Bank Branch, and External Bank Account are business objects whose details are stored in non-HCM tables. However, you can load them using HCM Data Loader. This topic describes aspects of the External Bank Account object that you must understand to load external bank accounts successfully.

Bank and Bank Branch

Before you can create an External Bank Account object, you must create both the bank and the bank branch where the bank account is held.

Loading External Bank Accounts

You supply external bank account data in the ExternalBankAccount.dat file for processing by HCM Data Loader. Multiple user keys are supported for referencing the external bank account. For details of these user keys, view the External Bank Account object on the Business Object Details page.

The External Bank Account Owner component identifies a person or third party to whom the account belongs. A bank account can be associated with multiple owners, of whom one must be the primary owner. Use the discriminator ExternalBankAccountOwner to load External Bank Account Owner components. Set the PrimaryFlag attribute of the relevant component to Y to identify the primary owner. Multiple user keys are supported for referencing the external bank account owner. For details of these user keys, view the External Bank Account Owner component on the Business Object Details page.

Note: You can't load external bank accounts for pending workers.
This example `ExternalBankAccount.dat` file creates an External Bank Account component and associates it with two owners, one of whom is the primary owner.

```plaintext
METADATA|ExternalBankAccount|BankNumber|BankBranchNumber|CountryCode|AccountNumber|IBAN|AccountName|CurrencyCode
MERGE|ExternalBankAccount|100001A|111111A|CA|12345678|CA 23 ANBK 3350 1234 5678 20|J and P Smith|USD

METADATA|ExternalBankAccountOwner|BankNumber|BankBranchNumber|CountryCode|AccountNumber|CurrencyCode|PersonNumber|PrimaryFlag
MERGE|ExternalBankAccountOwner|100001A|111111A|CA|12345678|USD|121011|Y
MERGE|ExternalBankAccountOwner|100001A|111111A|CA|12345678|USD|126231|N
```

**Note:** You can only load `ExternalBankAccount` in single threaded mode, even if you provide multiple load threads.

### Inactivating External Bank Accounts

You can delete an External Bank Account object, and recreate it if necessary, using HCM Data Loader. Alternatively, you can make the bank account inactive. This example `ExternalBankAccount.dat` file makes an external bank account inactive by including the `InactiveFlag` attribute and setting it to `Y`.

```plaintext
METADATA|ExternalBankAccount|BankNumber|BranchNumber|CountryCode|CurrencyCode|AccountNumber|InactiveFlag
MERGE|ExternalBankAccount|100001A|111111A|CA|USD|12345678|Y
```

### Example of Loading Organization Payment Methods

An organization payment method defines the payment methods for a legislative data group. Payment methods specify a payment type, such as check or direct deposit, a source bank account, and other details.

Using HCM Data Loader, load at least one organization payment method for each combination of legislative data group, payment type, and currency.

To load an organization payment method, you define at least one payment source. If you load multiple payment sources, then you can create payment rules. The payment rules determine which payment source to use for each TRU and also validate or process the distribution of payments. Create payment method rules with effective start dates on or before the start dates of other objects that refer to your payment method rule.

In this example, you create an Organization Payment Method and define payment sources and payment rules.

1. Create an Organization Payment Method `HDL_OPM_TRU` for the Vision Corporation US LDG LDG.
2. Load the two payment sources to associate bank accounts and other sources of funds with it.
3. Define the payment rules to determine the appropriate payment source based on tax reporting unit.

This figure shows the steps to create an Organization Payment Method.
As this image shows, for your organization payment method, you create the payment sources and rules.

**Create an Organization Payment Method**

These payment method lines create an organization payment method **HDL_OPM_TRU**:

- **HDL_OPM_TRU**
  - **METADATA | OrganizationPaymentMethod**
    - **SourceSystemOwner**: Vision Corporation US LDG
    - **SourceSystemId**: HDL_OPM_TRU
    - **LegislativeDataGroupName**: Vision Corporation US LDG
    - **EffectiveStartDate**: 2018/01/01
    - **OrganizationPaymentMethodName**: HDL Organization Payment Method with TRU rule
    - **OrganizationPaymentMethodCode**: HDL_OPM_TRU
    - **CurrencyCode**: USD
    - **PaymentTypeCode**: EFT

**Create Payment Sources**

These DAT files to load the two payment sources:

<table>
<thead>
<tr>
<th>Payment Source</th>
<th>Use To</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDL_OPM_TRU_PS_VISION</td>
<td>Pay workers or third-party people based in the Vision Corporation tax reporting unit.</td>
</tr>
<tr>
<td>HDL_OPM_TRU_PS_DFLT</td>
<td>Pay workers in all other tax reporting units. Set this payment source as the default payment source.</td>
</tr>
</tbody>
</table>

**METADATA | PaymentSource**
- **SourceSystemOwner**: Vision Corporation US LDG
- **SourceSystemId**: HDL_OPM_TRU
- **LegislativeDataGroupName**: Vision Corporation US LDG
- **EffectiveStartDate**: 2018/01/01
- **OrganizationPaymentMethodId**: HDL_OPM_TRU
- **PaymentSourceCode**: HDL_DEFLT
- **PaymentSourceName**: HDL Payment Source Corporate Payroll Account-202
- **BankAccountName**: 808123663
- **BankReferenceEFT**: 456102113
- **CompanyReferenceEFT**: 456102113

**MERGE | PaymentSource**
- **VISION | HDL_OPM_TRU_PS_DFLT**
- **Vision Corporation US LDG | 2018/01/01 | HDL_OPM_TRU | HDL_DEFLT | HDL Payment Source Corporation Payroll Account-202 | 808123663 | 456102113

**MERGE | PaymentSource**
- **VISION | HDL_OPM_TRU_PS_VISION**
- **Vision Corporation US LDG | 2018/01/01 | HDL_OPM_TRU | HDL_VISION | HDL Payment Source Vision | Vision Nacha Account-202 | 234506332 | 567890123
Create Payment Rules

Finally, use the DAT file to load these two payment rules.

<table>
<thead>
<tr>
<th>Payment Rule</th>
<th>What It Does</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDL_OPM_TRU_DFLT</td>
<td>Sets the payment source HDL_OPM_TRU_PS_DFLT as the default source.</td>
</tr>
<tr>
<td>HDL_OPM_TRU_VISION</td>
<td>Associates the payment source HDL_OPM_TRU_PS_VISION with the Vision Corporation TRU.</td>
</tr>
</tbody>
</table>

**Note:** When updating these existing components, you can’t change either the first effective start date or last effective end date:
- Organization payment method
- Payment source
- Payment method rule

**Related Topics**
- Organization Payment Methods
- How Payment Methods and Payroll Definitions Work Together
- Setting Up Payment Sources in Organization Payment Methods Example

Example of Loading Personal Payment Methods

Use HCM Data Loader to load payment method details for a person. Personal payment methods associate people to specific payment method, currency, and payment source.

In this example, Jane signed up for direct deposit of her pay check. Effective January 1, 2016, you create a payment method for her.
As this image shows, loading a personal payment method is a four step process.

### Prerequisites

1. Associate Employee With Payroll
2. Load Organization Payment Method
3. Load Organization Payment Method
4. Load Bank Account Details For Employee

Before creating a personal payment method, do these tasks:

- Associate Jane with a payroll.
- Load the organization payment method.
- Create the bank account the personal payment method will pay into.

As Jane’s payment type is **Direct Deposit**, load these bank account details for her account:

<table>
<thead>
<tr>
<th>Bank Account</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Number</td>
<td>1265896441</td>
</tr>
<tr>
<td>Bank Name</td>
<td>Bank of America</td>
</tr>
<tr>
<td>Routing Transit Number (Bank Branch Number)</td>
<td>122122125</td>
</tr>
</tbody>
</table>

Use the **PersonalPaymentMethod.dat** file to create a personal payment method. This file deposits only the first $10000 into the bank account.

```
METADATA|PersonalPaymentMethod|LegislativeDataGroupName|AssignmentNumber|PersonalPaymentMethodCode|EffectiveStartDate|PaymentAmountType|Amount|ProcessingOrder|OrganizationPaymentMethodCode|Percentage|BankName|BankBranchNumber|BankCountryCode|BankAccountNumber
MERGE|PersonalPaymentMethod|Vision Corporation US LDG|234299A|PPM3_HDL_NO_PO|2016/01/01|M|10000||PM US Sun Power Check||Bank of America|511000025|US|1265896441
```

**Related Topics**

- How Payment Methods and Payroll Definitions Work Together
Example of Loading Third-Party Organization Payment Method

Create payment methods to make payments to external organizations, such as pension providers or professional bodies. These payment methods include details, such as the type of payment and the bank account details.

Example

This image shows how you associate a third-party organization representing a Texas Court to a payment method. This enables the payment of child support deductions that are deducted through the payroll to be paid to the court. Also, you add a calculation card to an employee for the involuntary deduction for the child support payment.
Before You Begin

Do these steps:

1. Create a third-party organization.
2. Load the Organization Payment Methods for your payroll.
3. Load the external bank account.

Load the External Bank Account

You can use the External Bank Account Owner component to identify the third-party to whom the account belongs to.

This example ExternalBankAccount.dat file creates an External Bank Account component and associates it with a third-party organization.

```
METADATA|ExternalBankAccount|BankNumber|BankBranchNumber|CountryCode|AccountNumber|IBAN|AccountName|CurrencyCode
MERGE|ExternalBankAccount|100001A|111111A|CA|12345678|CA23 ANBK 3350 1234 5678 20|J and P Smith|USD

METADATA|ExternalBankAccountOwner|BankNumber|BankBranchNumber|CountryCode|AccountNumber|CurrencyCode|
ThirdPartyNumber|PrimaryFlag
MERGE|ExternalBankAccountOwner|100001A|111111A|CA|12345678|USD|68299|Y
```

Use this dat file to associate a payment method with the third-party organization HDL_Texas_ORG and organization payment method Direct Deposit. Additionally, as the payment type is Direct Deposit, you load these bank account details.

```
<table>
<thead>
<tr>
<th>Bank Account Component</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Number</td>
<td>1265896441</td>
</tr>
<tr>
<td>Bank Name</td>
<td>Bank of America</td>
</tr>
<tr>
<td>Bank Branch Name</td>
<td>New York</td>
</tr>
<tr>
<td>Routing Transit Number</td>
<td>511000025</td>
</tr>
<tr>
<td>Account Type</td>
<td>Checking</td>
</tr>
</tbody>
</table>
```

```
METADATA|ThirdPartyOrganizationPaymentMethod|EffectiveStartDate|EffectiveEndDate|LegislativeDataGroupName|TimeDefinitionCode|PartyNumber|BankName|BankBranchNumber|BankAccountNumber|BankCountryCode|OrganizationPaymentMethodCode
MERGE|ThirdPartyOrganizationPaymentMethod|2016/01/01||USA LDG||68299|Bank of America|511000025|1265896441|US|OPM_US_Nacha
```

Related Topics

- How Payment Methods and Payroll Definitions Work Together
Example of Loading Third-Party Personal Payment Method

Use HCM Data Loader to create payment methods to people who aren’t on your payroll. When loading such payment methods, associate the third-party persons directly with the employee whose pay is subjected to deduction by supplying the assignment number. A third-party person can receive payments from more than one employee too.

Example

John Smith is an employee at your organization. Mary Smith is his former spouse and receives a child-support payment for each payroll period that’s deducted from John’s salary. You want to set up payments for Mary.

Before You Begin

Do these steps:

1. Create Mary as a third-party person.
2. Create the external bank account details.

Then, you create Mary’s third-party payment method and associate the payroll relationship for John by supplying a reference to his assignment. Finally, add the child support to John’s involuntary deduction calculation card and supply Mary’s name as Order Amount Payee.
This figure shows how you can pay a third-party person using payment methods and calculation cards.

**Prerequisites**
- Create Mary as third-party person
- Create the external bank account

**Third-Party Person Payment Method**
- Create Mary’s third-party payment method
- Associate Payroll Relationship for John

**Processing of Payment Methods in Calculation Cards**
- John’s Involuntary Deduction Card
- Add Child Support
- Supply Mary’s name in the Order Amount Payee

Use this example dat file to associate a payment method with the third-party person Mary Smith. The payment type for Mary is **Check**. The association with John’s payroll relationship is achieved by supplying a reference to his assignment.

```plaintext
METADATA|ThirdPartyPersonalPaymentMethod|EffectiveStartDate|LegislativeDataGroupName|OrganizationPaymentMethodCode|AssignmentNumber|PartyNumber|BankName|BankBranchNumber|BankCountryCode|BankAccountNumber
MERGE|ThirdPartyPersonalPaymentMethod|2016/01/01|USA LDG|Vision US1|E95516|395077|HSBC|568793|US|12345678
```

**Related Topics**
- How Payment Methods and Payroll Definitions Work Together
- Creating Third Parties Options
Overview of Loading Payroll Costing

Using HCM Data Loader, you can create payroll costing setup information for the different payroll accounts. For example, cost, offset, suspense, default, payroll liability, cash clearing, and cash accounts. You can define costing at various levels, such as Payroll, Element Eligibility, Department, Job, Position, Person, and Payment Source.
This figure shows the different levels of the costing hierarchy.

You enter the account information and any overrides for the different levels of the cost hierarchy. The application builds the account number, starting with the lowest level (element entry) of the cost hierarchy and ending with the highest level (payroll). It checks each level sequentially until it finds a value. If it finds an invalid cost combination, it places the costing result in a suspense account.
Before You Begin

To load payroll costing records, do these tasks:

- Create Cost Allocation key flexfield. This flexfield creates a structure to capture the account codes used to create accounting entries and to track and report labor costs.
- Enable cost hierarchy with levels to support each cost allocation key flexfield cost account segment.
- Assign cost Allocation key flexfield to a legislative data group.

Payroll Costing Accounts

Load accounts to cost payroll run and payment results, and to store invalid and accounts that are not allocated.

Using HCM Data Loader, you can do all of these tasks.

- Load cost accounts that store expenses and employer liabilities and charges.
- Create overrides by entering cost account numbers at lower levels of the cost hierarchy.
- Use priority accounts to cost elements that require the same account combination. For example, you use a priority account for an hourly earning element for laboratory work that is charged to a grant fund.
- Allocate a cost to more than one account by creating several accounts for an object. Specify the percentage to charge to each account. For example, allocate costs to split salary costs for a job shared between two cost centers.
- Load Offset accounts that create balancing entries required for double-entry bookkeeping.
- Set up suspense and default accounts at the payroll level and override them if required, with suspense and default accounts at the department level.

Related Topics

- Payroll Cost Allocation Key Flexfield Setup
- How Payroll Costing Components Work Together
- Payroll Setup Tasks for Costing Accounts

Example of Loading Element Eligibility Costing

Use HCM Data Loader to assign costing information at the element level. You cost the element’s eligibility records when the results for the element affect net pay or employer liabilities. When costing an element, you specify the values to cost, the type of costing to use, and then you offset the cost entries.

You can load the priority account that specifies the accounting at the element level. For example, your company funds the entire hourly earning element for work performed in a lab from a single account. In such a case, you can create a priority account and specify 100 for the percentage.

Let’s suppose you cost a value for the same segment at higher levels in the costing hierarchy. In this case, though the value is defined at the element eligibility level, the application considers the higher segment value.
Example

The Payroll Manager of the Vision Corp organization wants to set up costing for the **Employer Union Pension Expense** element and distribute its costs to the earning elements of a distribution group.

**Before You Begin**

Before loading element costing, ensure that you do these steps:

1. Set up the Cost Allocation key flexfield.
2. Create costing for element eligibility records for each of the pensionable earnings elements, such as the regular wages and overtime wages.
3. Create the distribution group, **Pensionable Wages** that carries the costs of the distributed element. This group includes the employee’s wage elements.

After setting up these prerequisites, you cost the employer portion of the pension liability by creating an **Employer Union Pension Expense** element.

For Distributed costing types, the calculation for distributed costing starts with the values loaded in the element eligibility costing record. The calculation derives values for other segments from the costing values on the associated entries of the distribution group. Further, it generates one costing result for each entry in the distribution group. Finally, you transfer the costing entries for the payroll run results to the General Ledger.

This figure shows how you cost details for an element.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Costing Type</td>
<td>Distributed</td>
</tr>
<tr>
<td>Transfer to General Ledger</td>
<td>Yes</td>
</tr>
</tbody>
</table>
When you are loading costing details for element eligibility, you must specify the costed input values.

Load these cost account values for segments that you don’t want the costing calculation to derive from distributed costing:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fund (Segment3)</td>
<td>5220 Employer Union Pension Expense account</td>
</tr>
<tr>
<td>Appropriation (Segment 4)</td>
<td>1001</td>
</tr>
<tr>
<td>Funding Source (Segment 5)</td>
<td>1010</td>
</tr>
</tbody>
</table>

Load these values for segments in Offset Accounts. Here, the offset account is the payable liability account, and the balance sheet account numbers are for the Division and Department segments:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Division (Segment1)</td>
<td>00</td>
</tr>
<tr>
<td>Department (Segment2)</td>
<td>000</td>
</tr>
<tr>
<td>Natural Account (Segment6)</td>
<td>2152</td>
</tr>
</tbody>
</table>

Related Topics
- Costing of Elements Options
• How Distributed Costing Is Calculated

• How Payroll Costing Components Work Together

Example of Loading Costing of Payroll

Use HCM Data Loader to assign account information to payrolls. You can select this highest level for segments that seldom change for the people assigned to the payroll, such as company and line of business. As a best practice, set up a suspense and default account at the payroll level. Otherwise, you must set up a suspense and default account for every department to ensure you charge invalid costs to an account.

Example

Vision Corp has multiple payrolls, such as weekly and monthly and loads costing details at the payroll level. And they use the costing information to report labor cost and generate journal entries for all their employees for the specified payroll.

Load these costing details for the monthly payroll Vision Corp Monthly.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fund (Segment1)</td>
<td>All Funds; 0001</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>Salaries Appropriation; 1001</td>
</tr>
<tr>
<td>Funding Source (Segment3)</td>
<td>Original Funding Source; 1010</td>
</tr>
<tr>
<td>Department (Segment4)</td>
<td>General Government Agencies; 1000</td>
</tr>
<tr>
<td>Organization (Segment5)</td>
<td>Payroll; 1003</td>
</tr>
<tr>
<td>Program (Segment6)</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Object (Segment7)</td>
<td>Unspecified; 0000</td>
</tr>
<tr>
<td>Project (Segment8)</td>
<td>Unspecified; 0000</td>
</tr>
</tbody>
</table>

Load values for both suspense and default accounts.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fund (Segment1)</td>
<td>Unspecified; 0000</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>Balance Sheet; 0000</td>
</tr>
</tbody>
</table>
### Example of Loading Costing of Departments

Use HCM Data Loader to define payroll costing rules at the Department level. For example, in a large enterprise you might set up default accounts for departments where the managers commonly review and resolve their department's expenses.

When you load costs at the department level, you can specify the percentage each account receives of the cost. If the total allocation isn't 100 percent, then the application places the invalid cost in a default account.

---

This example .dat file loads the costing details for the monthly payroll.

**Related Topics**

- How Payroll Costing Components Work Together
- Payroll Setup Tasks for Costing Accounts
- Payroll Cost Allocation Key Flexfield Setup
Example

Vision Corporation’s Payroll Manager wants to allocate costing at the department level to two cost centers. As this figure shows, she allocates 35 percent of the costs of the Administration department to the Eastern District Office. She allocates the remainder to the Western District Office.

The Payroll Manager loads these costing details for the Administration department to two district officers as below:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Eastern District Office</th>
<th>Western District Office</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>35</td>
<td>65</td>
</tr>
<tr>
<td>Fund (Segment1)</td>
<td>Pooled Cash Fund: 0900</td>
<td>Pooled Cash Fund: 0901</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>1001</td>
<td>1001</td>
</tr>
<tr>
<td>Funding Source (Segment3)</td>
<td>1010</td>
<td>1011</td>
</tr>
</tbody>
</table>

Further, she uses the payroll suspense account to store costed payroll run results and prepayment results with invalid account combinations. She loads these suspense account details.
As Vision Corporation’s department managers review and resolve their department expenses, the Payroll Manager sets up default accounts for the Administration department. She loads these values for the default account.

This example dat file loads the costing details for the two departments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fund (Segment1)</td>
<td>All Funds; 0001</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Funding Source (Segment3)</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Department (Segment4)</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Organization (Segment5)</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Object (Segment7)</td>
<td>Unspecified; 0000</td>
</tr>
<tr>
<td>Project (Segment8)</td>
<td>Unspecified; 0000</td>
</tr>
</tbody>
</table>

This example dat file loads the costing details for the two departments.

METADATA|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|DepartmentName|LegislativeDataGroupName
MERGE|CostAllocationV3|4712/12/31|2010/01/01|ORG|Administration|Vision Corp
METADATA|CostAllocationAccountV3|SourceType|DepartmentName|LegislativeDataGroupName|EffectiveDate|SourceSubType|Proportion|SubTypeSequence|ConcatenatedSegment|Segment1|Segment2|Segment3|Segment4|Segment5|Segment6|Segment7|Segment8
MERGE|CostAllocationAccountV3|ORG|Administration|Vision Corp|2010/01/01|COST|0.35|1||0900|1001|1010||
MERGE|CostAllocationAccountV3|ORG|Administration|Vision Corp|2010/01/01|COST|0.65|2||0901|1333|220||
MERGE|CostAllocationAccountV3|ORG|Administration|Vision Corp|2010/01/01|SUSP|1|1|0001.0000.0000.0000.0000.0000.0000.0000|
Example of Loading Costing of Jobs

Use HCM Data Loader to define payroll costs based upon the job category, such as cost centers or project funds. Let’s assume that you want to compare cost results for jobs in different cost centers. Then, you allocate costs at the job level to generate the cost results based on the job.

You can load cost account segments to override those segments defined at other levels. If you divide costs among accounts, you can specify the percentage each account receives of the cost. If the total allocation isn’t 100 percent, then the application places the costs that are not allocated to a default account.

Example

Assume that you want to cost accounts to store expenses and employer liabilities and charges for the job Senior Manager. This table shows the job details that you want to cost for the employee.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>100</td>
</tr>
<tr>
<td>Fund (Segment1)</td>
<td>Pooled Cash Fund: 0900</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>1001</td>
</tr>
</tbody>
</table>

Use these data files to load the cost segments for the job Senior Manager:

```plaintext
MERGE|CostAllocationAccountV3|ORG|Administration|Vision Corp|2010/01/01|DFLT|1|1|
0000.0000.0000.0000.0000.0000.0000.0000 ||||| |
```

Related Topics

- Allocating Costs to Accounts
- Payroll Setup Tasks for Costing Accounts

Use these data files to load the cost segments for the job Senior Manager:

```plaintext
MERGE|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|JobCode|SetCode|LegislativeDataGroupName
MERGE|CostAllocationV3|4712/12/31|2010/01/01|JOB|Senior Manager|COMMON|Vision Corp
MERGET|CostAllocationAccountV3|SourceType|JobCode|SetCode|LegislativeDataGroupName|EffectiveDate|
SourceSubType|Proportion|SubTypeSequence|Segment1|Segment2
MERGE|CostAllocationAccountV3|JOB|Senior Manager|COMMON|Vision Corp|2010/0101|COST|1|0900|1001
```

Related Topics

- Payroll Setup Tasks for Costing Accounts
- Allocating Costs to Accounts
- How to Allocate Costs to Several Accounts
Example of Loading Costing of Positions

If you’re using position management at your enterprise to track the cost of turnover to the enterprise, then you allocate costs at the position level. For example, you want to compare cost results for positions in different cost centers. So, you allocate the costs at the position level to generate cost results based on the position.

Let’s consider this example. Vision Corp uses position management and wants to cost these account details for the position Senior Product Manager in Utilities business unit.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>100</td>
</tr>
<tr>
<td>Fund (Segment1)</td>
<td>Government Fund; 1000</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Funding Source (Segment3)</td>
<td>Funding Source; 2000</td>
</tr>
</tbody>
</table>

Use this dat file to cost the position details.

METADATA|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|PositionCode|LegislativeDataGroupName|BusinessUnitName
MERGE|CostAllocationV3|4712/12/31|2010/01/01|POS|Senior Product Manager|Vision Corp|Utilities
METADATA|CostAllocationAccountV3|SourceType|PositionCode|LegislativeDataGroupName|EffectiveDate|SourceSubType|Proportion|SubTypeSequence|Segment1|Segment2|Segment3|BusinessUnitName
MERGE|CostAllocationAccountV3|POS|Senior Product Manager|Vision Corp|2010/01/01|COST|1|1|100|0000|2000|Utilities

Related Topics

- Payroll Setup Tasks for Costing Accounts
- Allocating Costs to Accounts
- How to Allocate Costs to Several Accounts

Example of Loading Payment Source Costing

Use HCM Data Loader to enter account information for your payment sources. Imagine a scenario where a delay exists between the date the payment is issued by using a cheque and the date the cheque is cleared. In such a case, your company wants to create a cash clearing account to track payments.

Using HCM Data Loader, you load a payment source and create liability, cash clearing, and cash accounts for it.

Note: You load the same account information that you use for the cash and cash clearing account that you created in the General Ledger.

This table summarizes the values that you load for the various accounts:
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Cash Account</th>
<th>Cash Clearing Account</th>
<th>Liability Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fund (Segment1)</td>
<td>All Funds; 0001</td>
<td>Pooled Cash Fund; 0900</td>
<td>All Funds; 0001</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>Salaries Appropriation; 1001</td>
<td>Salaries Appropriation; 1002</td>
<td>Salaries Appropriation; 1002</td>
</tr>
<tr>
<td>Funding Source (Segment3)</td>
<td>Original Funding Source; 1010</td>
<td>Funding Source; 1999</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Department (Segment4)</td>
<td>Progress Top Level; 0001</td>
<td>General Government Agencies; 1000</td>
<td>Balance Sheet; 0000</td>
</tr>
<tr>
<td>Organization (Segment5)</td>
<td>Administration; 1001</td>
<td>Administration; 1001</td>
<td>Payroll; 1003</td>
</tr>
<tr>
<td>Program (Segment6)</td>
<td>Programs; 0001</td>
<td>Programs; 0001</td>
<td>Legislative; 1100</td>
</tr>
<tr>
<td>Object (Segment7)</td>
<td>Cash; 11110</td>
<td>Object; 11140</td>
<td>On Account Cash; 11150</td>
</tr>
<tr>
<td>Project (Segment8)</td>
<td>Administrative Projects; 100000</td>
<td>Capital Project; 100001</td>
<td>Broadway Extension; 100004</td>
</tr>
</tbody>
</table>

This dat file loads these account values for the payment source Vision Corp Check PS in Vision Corp Check PM payment method:

```
MERGE|CostInfoV3|4712/12/31|2010/01/01|PM|Vision Corp Check PS|Vision Corp Check PM|Vision Corp|Y|Y|Y|||
METADATA|CostInfoV3|EffectiveEndDate|EffectiveStartDate|SourceType|PaymentSourceCode|OrganizationPaymentMethodCode|LegislativeDataGroupName|CostableType|CostedFlag|TransferToGlFlag|GUID|SourceSystemId|SourceSystemOwner
MERGE|CostAllocationAccountV3|2010/01/01|CLNRG|1100.0003.1000.1001.0001.11140.100001|||
METADATA|CostAllocationAccountV3|EffectiveDate|SourceSubType|Proportion|SubTypeSequence|ConcatenatedSegment| GUID|SourceSystemId|SourceSystemOwner
MERGE|CostAllocationAccountV3|2010/01/01|CASH|1100.0003.1000.1001.0001.11140.100001|||
METADATA|CostAllocationAccountV3|EffectiveDate|SourceSubType|Proportion|SubTypeSequence|ConcatenatedSegment| GUID|SourceSystemId|SourceSystemOwner
```

**Related Topics**
- Payroll Setup Tasks for Costing Accounts
- How Payroll Cost Results are Calculated
- Payroll Setup Tasks for Transferring Costs to General Ledger

**Example of Loading Person Costing**

Use HCM Data Loader to manage costing at the person level to track costs for people in your enterprise. You can either cost all the elements that the person is eligible to receive or cost individual elements. You allocate costs to single or multiple accounts.
Let’s suppose you start a new project and want to track the costs incurred by the employees reassigned temporarily to the project. To monitor these costs, you can set up costing at the person level for these employees.

When you load costs at the person level, you can specify the percentage each account receives of the cost. If the total allocation isn’t 100 percent, then the application places the cost that is not allocated in a default account.

Using HCM Data Loader, you can load costing for a person at the assignment and payroll relationship levels. Further, you can allocate costing for an element at the assignment or payroll relationship level for a person.

### Example of Loading Person Costing at Payroll Relationship Level

Let’s consider this example where you cost Vijay’s elements to the Special Revenue Fund. When you define the costs at the payroll relationship level, you need not define the costs at each element level for these segments.

Load these details to allocate costs for Vijay (Payroll Relationship Number: 12345678) at the payroll relationship level.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>100</td>
</tr>
<tr>
<td>Fund (Segment1)</td>
<td>Special Revenue Fund (1100)</td>
</tr>
<tr>
<td>Appropriation (Segment2)</td>
<td>Salary Appropriation (1002)</td>
</tr>
<tr>
<td>Funding Source (Segment3)</td>
<td>2000</td>
</tr>
<tr>
<td>Department (Segment4)</td>
<td>2000</td>
</tr>
<tr>
<td>Organization (Segment5)</td>
<td>Payroll (1003)</td>
</tr>
</tbody>
</table>

This example dat file loads the costing details for the person at the payroll relationship level.

```
METADATA|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|PayrollRelationshipNumber|LegislativeDataGroupName
MERGE|CostAllocationV3|4712/12/31|2010/01/01|PREL|12345678|Vision Corp
METADATA|CostAllocationAccountV3|SourceType|PayrollRelationshipNumber|LegislativeDataGroupName|EffectiveDate|SourceSubType|Proportion|SubTypeSequence|Segment1|Segment2|Segment3|Segment4|Segment5
MERGE|CostAllocationAccountV3|PREL|12345678|Vision Corp|2010/01/01|COST|1|1|1100|1002|2000|2000|1003
```

### Example of Loading Person Costing at Assignment Level

As Vision Corp encourages employees to work across multiple projects, employee’s costs should be allocated in the same way. In this example, you allocate costing at the assignment level for a person who divides the time worked between two managers at different cost centers. Further, you override costing for a specific element at the assignment level.
Lynda Jones writes advertisements for the Advertising division. For the next few months, she will spend 40 percent of his time in designing customer preference surveys for the Market Research division. Now, you split Lynda's cost between Advertising and Market Research divisions, which belong to different cost centers.

Assign these percentages to the two departments:

<table>
<thead>
<tr>
<th>Percent</th>
<th>Department</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>3180 (Market Research)</td>
</tr>
<tr>
<td>60</td>
<td>3190 (Advertising)</td>
</tr>
</tbody>
</table>

The assignment number of Lynda is 12345.

This example .dat file loads the costing details for a person at the assignment level:

```
METADATA|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|AssignmentNumber|
LegislativeDataGroupName
MERGE|CostAllocationV3|4712/12/31|2010/01/01|ASG|12345|Vision Corp
METADATA|CostAllocationAccountV3|SourceType|AssignmentNumber|LegislativeDataGroupName|EffectiveDate|
SourceSubType|Proportion|SubTypeSequence|Segment4
MERGE|CostAllocationAccountV3|ASG|12345|Vision Corp|2010/01/01|COST|0.4|1|3180
MERGE|CostAllocationAccountV3|ASG|12345|Vision Corp|2010/01/01|COST|0.6|2|3190
```

**Example of Loading Person Element Costing at Assignment Level**

In this example, all elements of the employee John go to the cost center Payroll and are defined at the payroll relationship level. However, in a particular period, the Vision Corp pays a bonus to John and the costs of the bonus should go to his home department, which is Finance.

Here, you define the Finance cost center for the Bonus element using the assignment costing element.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element Name</td>
<td>Vision_Corp_Bonus</td>
</tr>
<tr>
<td>Percentage</td>
<td>100</td>
</tr>
<tr>
<td>Funding Source</td>
<td>1010</td>
</tr>
<tr>
<td>Department</td>
<td>3120 Finance Department</td>
</tr>
</tbody>
</table>

The employee’s assignment number is 12345 and the Payroll Relationship number is 12345678.

This .dat file loads the costing details for the two departments.

```
METADATA|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|AssignmentNumber|
PayrollRelationshipNumber|ElementCode|LegislativeDataGroupName
```
Example of Loading Person Element Costing at Payroll Relationship Level

In this example, you update the funding source for the ANC_ELE_COMP_VISION Entitlement element of Mandy Steward as University Grants for research. The application derives the funding source and department for all other elements from element eligibility costing and department level costing. Then, you can define these segments at payroll relationship element level.

For the ANC_ELE_COMP_VISION Entitlement element, load these costs at the payroll relationship level for Mandy.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>100</td>
</tr>
<tr>
<td>Funding Source</td>
<td>University Grants for research 2003</td>
</tr>
<tr>
<td>Department</td>
<td>Education 2110</td>
</tr>
<tr>
<td>Organization</td>
<td>Payroll (1003)</td>
</tr>
</tbody>
</table>

The employee’s payroll relationship number is 12345678.

This example .dat file loads the costing details at the payroll relationship level for the person.

Example of Loading Element Entry Costing

David works for Vision Corp and all of his elements go to his home department Payroll (3001). However, last Monday, David has provided services to HR department (3002) for 8 hours. Now, these hours should be costed to the HR department. The services are categorized as consulting (4001).

Using HCM Data Loader, load these element entry costing values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective Start Date</td>
<td>01-Jan-2011</td>
</tr>
</tbody>
</table>
The employee’s assignment number is 12345 and the payroll relationship number is 12345678.

This example .dat file loads the element entry level costing details for David.

```
METADATA|CostAllocationV3|EffectiveEndDate|EffectiveStartDate|SourceType|AssignmentNumber|ElementCode|
MultipleEntryCount|LegislativeDataGroupName|EntryType
MERGE|CostAllocationV3|4712/12/31|2010/01/01|EE|12345|Hourly Salary|1|Vision Corp|E
METADATA|CostAllocationAccountV3|SourceType|AssignmentNumber|ElementCode|MultipleEntryCount|
LegislativeDataGroupName|EffectiveDate|SourceSubType|Proportion|SubTypeSequence|Segment4|Segment6|EntryType
MERGE|CostAllocationAccountV3|EE|12345|Hourly Salary|1|Vision Corp|2010/01/01|COST|1|1|3002|4001|E
```

**Related Topics**

- Payroll Cost Allocation Key Flexfield Setup
- How Payroll Costing Components Work Together
- Payroll Setup Tasks for Costing Accounts
18 Using Transformation Formulas

HCM Data Loader Transformation Formula

Overview

Often times, your existing data or the data that you upload might not be in the format recognized by HCM Data Loader. In such cases, use the HCM Data Loader Transformation formula to transform your data into a format that is supported by HCM Data Loader.

Let’s consider these examples.

- An inbound file contains data that needs to be loaded using different business objects in HCM Data Loader. Here, the content of the file needs to be split across more than one HCM Data Loader file.
- You might create a transformation formula to convert an attribute in the file to another attribute that you derive using value sets.
- You want to change a person number into an assignment number. In this case, you will use a more complex formula to convert the attributes.

Load Data From File Flow

You use the Load Data From File flow to calls the HCM Data Loader transformation formula. This formula supports these capabilities.

- Creates transformed files in UCM.
- Handles multiple objects in a single load.
- Support flexfield attributes and source keys.

And it must contain values for the file name, file discriminator, and business operation. You can potentially load different types of data, using different file discriminators and file names with the supported business operations.

This example specifies the file name in the formula as `PersonalPaymentMethod`, the file discriminator as `PersonalPaymentMethod`, and the business operation as MERGE.

```/*HDL Related Outputs*/
FileName = 'PersonalPaymentMethod'
BusinessOperation = 'MERGE'
FileDiscriminator = 'PersonalPaymentMethod'
```

To view details about the file name, file discriminator, and a list of supported business operations, use the View Business Objects task in the Data Exchange work area.

1. On the View Business Objects page, search for and select your business object. In this example, the business object is Personal Payment Method.
2. On the Component Details page, you can find the name of the file, and the file discriminator and a list of supported actions for the object.
How You Transform Data Using HCM Data Loader Transformation Formula

Use the Load Data From File flow to transform data in the source file into a format that is supported by HCM Data Loader. You can submit this flow independently or include it in a flow that you create for automating data loads on a periodic basis. When you submit the flow, either manually or using a web service, you must specify a transformation formula to transform the data, as needed.

The flow contains these two tasks that help you to transform data into a HCM Data Loader format:

- Generate Data Loader File
- Initiate Data Loader Task

As this figure shows, the first step is to submit the Load Data From File flow. This flow takes data from the flat file and generates an equivalent file format for the data present in the input file.

Perform these steps to transform data using HCM Data Loader transformation formula:

1. From the Checklist or Data Exchange work area, submit the Load Data From File flow pattern.
2. The flow invokes the HCM Data Loader Transformation formula for the Content ID. Typically, you create HCM Data Loader Transformation formulas on the Manage Fast Formulas page. The type of the formula should be HCM Data Loader.
3. The Generate Data Loader File task reads the data file line by line, producing an equivalent HCM Data Loader format for each line. Finally, it creates a compressed file of all of the transformed data files and uploads to the UCM. Also, the task records the UCM Content ID.
4. The **Initiate Data Loader** task takes the UCM Content ID for the file generated by the **Generate Data Loader File** task. And it invokes the HCM Data Loader. The HCM Data Loader creates the required data in the HCM Cloud.

**How To Create Program for Automation**

You can submit the **Load Data from File** flow by using a web service.

As this figure shows, your program uploads the source file to content server, and retrieves the content ID for the Flow Actions Service web service. Then, the program calls the Flow Actions web service by supplying certain parameters.

When calling the web service, your program supplies these parameters.

- Name of the flow pattern, which is **Load Data from File**
- Content ID of the uploaded file
- Unique name to identify the flow instance being submitted
- Process configuration group name for special processing (optional)
- Transformation formula name (mandatory)

For more information about the Flow Actions Service web service, refer to the SOAP Web Services for Oracle HCM Cloud guide. For examples of its usage for automating file uploads, refer to the attachment for HCM Data Loader User Guide (1664133.1) on My Oracle Support at https://support.oracle.com.
Submit The Load Data From File Flow

From the Checklist or Data Exchange work area, use the Load Data from File flow pattern to transform data in your source file into the HDL format.

Assumptions

This procedure has these assumptions.

- You have the Human Capital Management Integration Specialist role.
- You have the source file ready to upload to Oracle WebCenter Content.
- If you have already uploaded the source file, you have the content ID handy.

Note: To upload files to the content server, browse to the source file on your file system, check it in to the content server, and retrieve its Content ID. For more information, see Oracle Fusion Middleware Using Oracle WebCenter Content guide.

Before You Begin

Before you submit the flow, ensure that you meet these prerequisites.

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>What You Should Do</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data transformation</td>
<td>If the data in the source file requires transformation, create and compile a transformation formula using the HCM Data Loader Transformation formula type. Use the Manage Fast Formulas task in the Payroll Calculation work area to do this compilation.</td>
</tr>
<tr>
<td>File encryption</td>
<td>Before loading encrypted files, ensure that the encryption keys exist for the secure file transfer. This process involves creating a service request, generating PGP key pairs, and sharing the encryption keys. Specify the Payroll Batch Loader Encryption Type parameter value for the process configuration group you select when running the flow or the web service. Valid values are PGPSIGNED, PGPUNSIGNED, and NONE. In the Setup and Maintenance work area, use the Manage Payroll Process Configuration task to specify processing parameters for your process configuration group.</td>
</tr>
<tr>
<td>Other processing parameters</td>
<td>In the Setup and Maintenance work area, use Manage Payroll Process Configuration task to add parameters for the process configuration group.</td>
</tr>
</tbody>
</table>

- Offering: Workforce Deployment
- Functional Area: Payroll
- Task: Manage Payroll Process Configuration
Characteristics | What You Should Do
--- | ---
Examples of processing parameters include Batch Error Mode, Logging Area, Logging Category, and Threads.

1. Click the **Submit a Payroll Flow** task.
2. In the **Legislative Data Group** option, select a legislative data group.
3. Search for and select the **Load Data from File** flow pattern.
4. Click **Next**.
5. Enter the parameters, as shown in this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payroll Flow</td>
<td>Descriptive name for this specific flow process.</td>
</tr>
<tr>
<td>Content Id</td>
<td>Enter the Content Id. The source file must already exists on the content server.</td>
</tr>
<tr>
<td>Transformation Formula</td>
<td>Select the required transformation formula. The type of the formula should be <strong>HCM Data Loader</strong>.</td>
</tr>
<tr>
<td>Process Configuration Group</td>
<td>Select your process configuration group.</td>
</tr>
</tbody>
</table>

6. On the Enter Parameters page, click **Next**.
7. On the Enter Flow Interaction page, click **Next**.
8. On the Schedule page, click **Next**.
9. On the Review page, click **Submit**.
10. In the confirmation dialog box, click **OK and View Checklist**.
11. On the Payroll Flow page, Task Details tab you should see a green check mark in the **Generate Data Loader File** and **Initiate Data Loader** rows, Task Type column. If not, on the toolbar, click the **Refresh** icon intermittently until you do.
13. On the Overview page, search for and click your payroll flow.
14. View the process results.
15. Check for any errors or warnings.

**Transformation Formula Input Variables**

Variables, such as **FileName**, **FileDiscriminator** and **LINEREPEATNO**, are available for all formulas of HCM Data Loader Transformation formula type. Additional variables may be available depending on the selected business object.

**Positions**

Using the transformation formula, you can assign attributes to the required position. Positions can range from 1 to N. Depending upon the business object, the positions can be either optional or mandatory.

Example:
This figure shows the different attributes for positions 2 through 8 for Balance Adjustments.

<table>
<thead>
<tr>
<th>LegislativeDataGroupName</th>
<th>BalAdjBatchId</th>
<th>SourceSystemOwner</th>
</tr>
</thead>
<tbody>
<tr>
<td>EffectiveDate</td>
<td>PayrollName</td>
<td>ConsolidationSetName</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SourceSystemId</td>
</tr>
</tbody>
</table>

Positions 2-8

In this example, you assign attributes to positions 2 through 8.

- POSITION2: EffectiveDate
- POSITION3: PayrollName
- POSITION4: LegislativeDataGroupName
- POSITION5: ConsolidationSetName
- POSITION6: BalAdjBatchId
- POSITION7: SourceSystemId
- POSITION8: SourceSystemOwner

FileName, FileDiscriminator, and BusinessOperation

These variables are required for all transformations.

- **FileName** is the name of the file for the business object.
- **FileDiscriminator** is the file discriminator for the business object.
- **BusinessOperation** refers to the operation, such as Merge or Delete that are performed by the HCM Data Loader process on the transformed file.

Here’s an example of values that you can supply for the input variables: FileName, FileDiscriminator, and BusinessOperation.

```plaintext
FileName = 'BalanceAdjustmentHeader'
BusinessOperation = 'MERGE'
FileDiscriminator = POSITION1
```

LINEREPEAT And LINEREPEATNO

**LINEREPEAT** allows a single line of input to be processed multiple times. And **LINEREPEATNO** indicates the number of repetitions.

For example, for time entry, there might be a regular time entry wage followed by a premium time entry wage.

**Example:** The Element Entry file contains these details.

```
Update|ElementEntryValue|Vision Corporation US LDG|WLM_Salary|2019/04/15|4712/12/31|E955160008191355-2|Amount|1002|2|E
Update|ElementEntryValue|Vision Corporation US LDG|WLM_Salary|2019/04/15|4712/12/31|E955160008191355-2|Amount|1003|3|E
```
The input line can be processed twice. The output file contains the element entry and element entry value as shown in this sample code snippet.

```
ELSE IF OPERATION='MAP' THEN
    (LegislativeDataGroupName=POSITION3
    ElementName=POSITION4
    EffectiveStartDate=POSITION5
    EffectiveEndDate=POSITION6
    AssignmentNumber=POSITION7
    InputValueName=POSITION8
    ScreenEntryValue=POSITION9
    MultipleEntryCount=POSITION10
    EntryType=POSITION11
    IF LINEREPEATNO=1 THEN
        (BusinessOperation='MERGE'
        BusinessObject='Element Entry'
        LINEREPEAT = 'Y'
        RETURN
        BusinessOperation,BusinessObject,AssignmentNumber,EffectiveEndDate,EffectiveStartDate,ElementName,EntryType,LegislativeDataGroupName,MultipleEntryCount,LINEREPEAT,LINEREPEATNO
    )
    ELSE
        (BusinessOperation='MERGE'
        BusinessObject='Element Entry Value'
        LINEREPEAT = 'N'
        RETURN
        BusinessOperation,BusinessObject,AssignmentNumber,EffectiveEndDate,EffectiveStartDate,ElementName,EntryType,LegislativeDataGroupName,MultipleEntryCount,InputValueName,ScreenEntryValue,LINEREPEAT,LINEREPEATNO
    )
```

**HCM Data Loader Transformation Formula Operations**

The transformation formula is invoked several times to derive different components that are required for processing the incoming data.

This table explains the various operations that you can do with the formula.

<table>
<thead>
<tr>
<th>Operation Type</th>
<th>Return Value</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILETYPE</td>
<td>OUTPUTVALUE</td>
<td>DELIMITED</td>
</tr>
<tr>
<td>DELIMITER</td>
<td>OUTPUTVALUE</td>
<td></td>
</tr>
<tr>
<td>READ</td>
<td>OUTPUTVALUE</td>
<td>NONE</td>
</tr>
<tr>
<td>MAP</td>
<td>Business Object Attributes</td>
<td>NA</td>
</tr>
<tr>
<td>NUMBEROFBUSINESSOBJECTS</td>
<td>OUTPUTVALUE</td>
<td>2</td>
</tr>
<tr>
<td>METADATALINEINFORMATION</td>
<td>METADATAn</td>
<td>METADATAn</td>
</tr>
</tbody>
</table>
Delimiter
The default delimiter that separates values is a pipe character. If your file uses a different delimiter, you must set the delimiter you want your formula.

This example specifies a comma character as the delimiter.

```/* Calculations */
IF OPERATION='FILETYPE' THEN
  OUTPUTVALUE='DELIMITED'
ELSE IF OPERATION='DELIMITER' THEN
  OUTPUTVALUE=','
END IF
```

Note: Ensure that the delimiter you enter in the formula is a single non-ASCII character and not part of any of the values to upload.

MAP
The MAP operation defines the return values related to a particular object. The return values must have the same names as the attributes specified in the application for that object.

For example, these RETURN values can be used to generate an Element Entry dat file.

```RETURN
  BusinessOperation,FileDiscriminator,FileName,AssignmentId,AssignmentNumber,CreatorType,DateEarned,EffectiveEndDate,Effecti...
```

Here, the `BusinessOperation` is set to MERGE and the `BusinessObject` is set to Element Entry.

METADATALINEINFORMATION
The application generates the file either with all defined attributes or with only specified attributes, depending on whether you specify a value for `METADATALINEINFORMATION` or not.

1. If you don’t specify a value for `METADATALINEINFORMATION`, then the application generates `METADATLINE` in the transformed file with all defined attributes for the business object.
2. If you specify a value, then the transformed file will contain only the attributes that you specified.

For the `METADATALINEINFORMATION` operation, you specify an array per business object being processed in the formula. The number of arrays should match the number specified in the `NUMBEROFBUSINESSOBJECTS` operation. The name of the array should be `METADATA` with the number as suffix. For example, `RETURN METADATA1, METADATA2` when the `NUMBEROFBUSINESSOBJECTS` is 2.

Note: The first two entries in the array are reserved to specify the `FileName` and `FileDiscriminator` of the business object.

Additionally, for `METADATALINEINFORMATION`, you can specify attributes with special characters for that business objects.

Notice that in this example `BalAdjBatchId(SourceSystemId)` has parenthesis.

If the file contains either Flexfield or SourceSystem references, then the application can’t resolve the default mapping of output parameter names and attributes.

Let’s consider this syntax: `jobEffSegment1(PER_JOBS_EIT_EFF=context)`. To allow this construct to be generated in the HCM Data Loader file, you define the `METADATA` line in the transformation formula. For each business object that appears in the output, you must define the `METADATA` content in an array.

Example:

```METADATA2[1] = 'Job' /*FileName*/```
METADATA2[2] = 'JobExtraInfo' /*FileDiscriminator*/
METADATA2[3] = 'EffectiveStartDate'
METADATA2[4] = 'EffectiveEndDate'
METADATA2[5] = 'JobCode'
METADATA2[7] = 'FLEX:PER_JOBS_EIT_EFF'
METADATA2[8] = 'EFF_CATEGORY_CODE'
METADATA2[9] = 'InformationType'
METADATA2[10] = 'JeiInformationCategory'
METADATA2[12] = 'SequenceNumber'
METADATA2[13] = 'jobEffSegment1(PER_JOBS_EIT_EFF=job-eff-context)'

Here's how the generated HCM Data Loader file looks like.

METADATA|JobExtraInfo|EffectiveStartDate|EffectiveEndDate|JobCode|SetCode|FLEX:PER_JOBS_EIT_EFF|
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EFF_CATEGORY_CODE</td>
<td>InformationType</td>
<td>JeiInformationCategory</td>
<td>LegislationCode</td>
<td>SequenceNumber</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jobEffSegment1(PER_JOBS_EIT_EFF=job-eff-context)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NUMBEROFBUSINESSOBJECTS**

This operation indicates the number of business objects being processed in the formula.

**HCM Data Loader Transformation Formula Return Values**

The return values for HCM Data loader formulas vary based on the business object and task action. They are the same as the attribute names and must include BusinessOperation, FileName, and FileDiscriminator.

Here's an example of return values.

/*Return Values*/
RETURN BusinessOperation,FileName,FileDiscriminator,Attribute1,Attribute2,...,Attributen
/*Attributes for a particular Business Object can be found from the View Business Objects UI under the HCM Data Loader task in the Data Exchange Work Area */

For the **NUMBEROFBUSINESSOBJECTS** and **METADATALINEINFORMATION** operations, the RETURN statement is as follows.

/*Return Values for NUMBEROFBUSINESSOBJECTS and METADATALINEINFORMATION Operation*/
IF OPERATION='FILETYPE' THEN
  OUTPUTVALUE='DELETED'
ELSE IF OPERATION='DELIMITER' THEN
  OUTPUTVALUE='|'
ELSE IF OPERATION='READ' THEN
  OUTPUTVALUE='NONE'
ELSE IF OPERATION = 'NUMBEROFBUSINESSOBJECTS' THEN(
  OUTPUTVALUE = '2'
  RETURN OUTPUTVALUE
)
ELSE IF OPERATION = 'METADATALINEINFORMATION' THEN
{
  METADATA1[1] = 'BalanceAdjustmentHeader' /*FileName*/ /*Reserved*/
  METADATA1[2] = 'BalanceAdjustmentHeader' /*FileDiscriminator*/ /*Reserved*/
  METADATA1[3] = 'LegislativeDataGroupName'
  METADATA1[4] = 'BatchName'
  METADATA1[5] = 'SourceSystemId'
  METADATA1[6] = 'SourceSystemOwner'

  METADATA2[1] = 'BalanceAdjustmentHeader' /*FileName*/ /*Reserved*/
  METADATA2[2] = 'BalanceAdjustmentGroup' /*FileDiscriminator*/ /*Reserved*/
  METADATA2[3] = 'EffectiveDate'
  METADATA2[4] = 'PayrollName'
  METADATA2[5] = 'LegislativeDataGroupName'
}
METADATA2[6] = 'ConsolidationSetName'
METADATA2[7] = 'BalAdjBatchId(SourceSystemId)'
METADATA2[8] = 'SourceSystemId'
METADATA2[9] = 'SourceSystemOwner'
RETURN METADATA1, METADATA2 /*Only two as Return value for NUMBEROFBUSINESSOBJECTS is 2*/

You can define variables with special characters. For example, use this structure to return the BalAdjBatchId(SourceSystemId) Source ID.

/*Return Values for the MAP Operation*/
FileName = 'BalanceAdjustmentHeader'
BusinessOperation = 'MERGE'
FileDiscriminator = POSITION1
EffectiveDate = POSITION2
PayrollName = POSITION3
LegislativeDataGroupName = POSITION4
ConsolidationSetName = POSITION5
"BalAdjBatchId(SourceSystemId)"= POSITION6
SourceSystemId = POSITION7
SourceSystemOwner = POSITION8

RETURN BusinessOperation, FileDiscriminator, FileName, EffectiveDate, PayrollName, LegislativeDataGroupName, ConsolidationSetName, "BalAdjBatchId(SourceSystemId)", SourceSystemId, SourceSystemOwner

/*Note BalAdjBatchId(SourceSystemId) is enclosed by double quotes while assigning value as well as while putting it in the return values list */

Sample Formula for HCM Data Loader File Transformation

In this example, the transformation formula specifies the transformation mechanism for an incoming comma separated delimited file. The formula’s return values are the same as the list of attributes for the personal payment method object.

Here’s the sample raw file for personal payment method, with comma as the delimiter.

2018/04/04,1,ZHRX_VS_US_TPPI_LDG_ONE,E955160008191423,ZHRX_VS_US_TPPI_Check,PPM1,M,10

And this code snippet has the formula for this example.

/*******************************
FORMULA NAME: Load Personal Payment Method
FORMULA TYPE: HCM Data Loader
**********************************/
/* Inputs */
INPUTS ARE OPERATION (text), LINENO (number), LINEREPEATNO (number),POSITION1 (text), POSITION2 (text),
POSITION3 (text), POSITION4 (text), POSITION5 (text), POSITION6 (text), POSITION7 (text), POSITIONS (text)

DEFAULT FOR POSITION1 IS 'NO DATA'
DEFAULT FOR POSITION2 IS 'NO DATA'
DEFAULT FOR POSITION3 IS 'NO DATA'
DEFAULT FOR POSITION4 IS '2'
DEFAULT FOR POSITION5 IS '100'
DEFAULT FOR POSITION6 IS 'NO DATA'
DEFAULT FOR POSITION7 IS 'NO DATA'
DEFAULT FOR POSITIONS IS 'NO DATA'
DEFAULT FOR LINEREPEATNO IS 1

IF OPERATION='FILETYPE' THEN
OUTPUTVALUE='DELIMITED'
ELSE IF OPERATION='DELIMITER' THEN
OUTPUTVALUE=','
ELSE IF OPERATION='READ' THEN
Sample HCM Data Loader Transformation Formula for Multiple Business Objects

In this example, the formula uses the user defined tables and personal payment method business objects. It converts the Person Number in the flat file into Assignment Number and uses the METADATALINEINFORMATION and NUMBEROFBUSINESSOBJECTS operations.

Here’s the sample of the raw input file.

| PPM|2018/04/04|1|ZHRX_VS_US_TFPI_LDG_ONE|955160008191423|ZHRX_VS_US_TFPI_Check|PPM1|M|10 |
| UDT|SM_UDT_4|Range|Number|Test UDT|USA LDG |

And this code snippet has the formula for this example.

```plaintext
/********************************************************** FORMULA NAME: Load User Defined Table and Personal Payment Method FORMULA TYPE: HCM Data Loader **********************************************************
/* Inputs */
INPUTS ARE OPERATION (text), LINENO (number), LINEREPEATNO (number), POSITION1 (text), POSITION2 (text), POSITION3 (text), POSITION4 (text), POSITION5 (text), POSITION6 (text), POSITION7 (text), POSITION8 (text), POSITION9 (text)
DEFAULT FOR POSITION1 IS 'NO DATA'
DEFAULT FOR POSITION2 IS 'NO DATA'
DEFAULT FOR POSITION3 IS 'NO DATA'
DEFAULT FOR POSITION4 IS '2'
DEFAULT FOR POSITION5 IS '100'
DEFAULT FOR POSITION6 IS 'NO DATA'
DEFAULT FOR POSITION7 IS 'NO DATA'
DEFAULT FOR POSITION8 IS 'NO DATA'
DEFAULT FOR POSITION9 IS 'NO DATA'
DEFAULT FOR LINEREPEATNO IS 1
IF OPERATION='FILETYPE' THEN
OUTPUTVALUE='DELIMITED'
ELSE IF OPERATION='DELIMITER' THEN
OUTPUTVALUE='|'
ELSE IF OPERATION='READ' THEN
OUTPUTVALUE='NONE'
```
ELSE IF OPERATION = 'NUMBEROFBUSINESSOBJECTS' THEN

| OUTPUTVALUE = '2'
| RETURN OUTPUTVALUE
|

ELSE IF OPERATION = 'METADATALINEINFORMATION' THEN

| METADATA1[1] = 'UserDefinedTable' /*FileName*/
| METADATA1[2] = 'UserDefinedTable' /*FileDiscriminator*/
| METADATA1[3] = 'UserTableCode'
| METADATA1[4] = 'RangeOrMatch'
| METADATA1[5] = 'UserKeyUnits'
| METADATA1[6] = 'UserRowTitle'
| METADATA1[7] = 'UserTableName'
| METADATA1[8] = 'LegislativeDataGroupName'
| METADATA2[1] = 'PersonalPaymentMethod' /*FileName*/
| METADATA2[2] = 'PersonalPaymentMethod' /*FileDiscriminator*/
| METADATA2[3] = 'EffectiveStartDate'
| METADATA2[4] = 'PersonalPaymentMethodCode'
| METADATA2[5] = 'AssignmentNumber'
| METADATA2[6] = 'Amount'
| METADATA2[7] = 'ProcessingOrder'
| METADATA2[8] = 'OrganizationPaymentMethodCode'
| METADATA2[9] = 'PaymentAmountType'
| METADATA2[10] = 'LegislativeDataGroupName'

RETURN METADATA1, METADATA2
|

ELSE IF OPERATION='MAP' THEN

IF POSITION1='UDT' THEN

| FileName = 'UserDefinedTable'
| BusinessOperation = 'MERGE'
| FileDiscriminator = 'UserDefinedTable'
| UserTableCode = POSITION2
| IF POSITION3='Range' THEN
| (RangeOrMatch = 'R')
| IF POSITION4='Number' THEN
| (UserKeyUnits = 'N')
| UserRowTitle = POSITION5
| UserTableName = POSITION2
| LegislativeDataGroupName = POSITION6
| RETURN BusinessOperation,FileDiscriminator,FileName,UserTableCode,RangeOrMatch,UserKeyUnits,UserRowTitle,UserTableName,LegislativeDataGroupName
|
| IF POSITION1='PPM' THEN

| FileName = 'PersonalPaymentMethod'
| BusinessOperation = 'MERGE'
| FileDiscriminator = 'PersonalPaymentMethod'
| EffectiveStartDate=POSITION2
| ProcessingOrder=POSITION3
| LegislativeDataGroupName=POSITION4
| AssignmentNumber=GET_VALUE_SET('SAMPLE_GET_ASG_NUM','|=PERSON_NUMBER='''||POSITION5||'''|')
| OrganizationPaymentMethodCode=POSITION6
| PersonalPaymentMethodCode=POSITION7
| PaymentAmountType=POSITION8
| Amount=POSITION9
| RETURN BusinessOperation,FileName,FileDiscriminator,EffectiveStartDate,PersonalPaymentMethodCode,AssignmentNumber,Amount,ProcessingOrder,LegislativeDataGroupName
|

ELSE
HCM Spreadsheet Data Loader

Overview

You can use HCM Spreadsheet Data Loader to load all payroll objects that HCM Data Loader supports. As the first step, you create a spreadsheet template for the required object from the Data Exchange Work area and further download the template in CSV format. The HCM Data Loader transformation formula transforms the raw delimited file to a format that suits the template.

This figure summarizes the process of transforming data that is uploaded using HCM Spreadsheet Data Loader.

Here's a summary of how the transformation process works.

1. From the Checklist or Data Exchange work area, submit the Load Spreadsheet Data from File flow pattern.
2. This flow pattern invokes the HCM Data Loader formula for the Content ID and has these tasks:

   a. Generate Data Loader File
   b. Initiate Spreadsheet Data Loader

3. The **Generate Data Loader File** task reads the data file line by line, producing an equivalent HCM Spreadsheet Data Loader format for each line. Finally, it creates a compressed file of all of the transformed data files and uploads it to UCM.

4. The **Initiate Spreadsheet Data Loader** task takes the compressed file generated by the **Generate Data Loader File** task and invokes the HCM Spreadsheet Data Loader. The HCM Spreadsheet Data Loader creates the required data in the HCM Cloud.

**Related Topics**

- Guidelines for Using HCM Spreadsheet Data Loader
- How Data Is Uploaded Using HCM Spreadsheet Data Loader
- HCM Spreadsheet Data Loader Templates
- Create and Edit Spreadsheet Templates
- Guidelines for Designing Spreadsheet Templates

**Sample HCM Spreadsheet Data Loader Formula**

In this example, the transformation formula specifies the transformation mechanism for an incoming pipe separated delimited file. The formula's return values are the same as the list of attributes in the template file for the User Defined Table business object.

And this code snippet has the formula for this example.

```/* Inputs */
INPUTS ARE OPERATION (text), LINENO (number), LINEREPEATNO (number), POSITION1 (text), POSITION2 (text),
POSITION3 (text), POSITION4 (text), POSITION5 (text), POSITION6 (text), POSITION7 (text), POSITION8 (text)
DEFAULT FOR POSITION1 IS 'NO DATA'
DEFAULT FOR POSITION2 IS 'NO DATA'
DEFAULT FOR POSITION3 IS 'NO DATA'
DEFAULT FOR POSITION4 IS 'NO DATA'
DEFAULT FOR POSITION5 IS 'NO DATA'
DEFAULT FOR LINEREPEATNO IS 1
IF OPERATION='FILETYPE' THEN
  OUTPUTVALUE='DELIMITED'
ELSE IF OPERATION='DELIMITER' THEN
  OUTPUTVALUE='|'
ELSE IF OPERATION='READ' THEN
  OUTPUTVALUE='NONE'
ELSE IF OPERATION = 'NUMBEROFBUSINESSOBJECTS' THEN
  (OUTPUTVALUE = '1'/*Always be 1*/
  RETURN OUTPUTVALUE
)
ELSE IF OPERATION = 'METADATALINEINFORMATION' THEN
  
  METADATA1[1] = 'SMUDT' /*TemplateName*/
METADATA1[2] = 'UserDefinedTable' /*FileDiscriminator*/
METADATA1[3] = 'UserDefinedTable_UserTableCode'
METADATA1[4] = 'UserDefinedTable_LegislativeDataGroupName'
```
METADATA1[5] = 'UserDefinedTable_UserTableName'
METADATA1[6] = 'UserDefinedTable_RangeOrMatch'
METADATA1[7] = 'UserDefinedTable_UserRowTitle'
METADATA1[8] = 'UserDefinedTable_UserKeyUnits'
RETURN METADATA1 /*You can return only one METADATA for the respective template*/

ELSE IF OPERATION='MAP' THEN
{
FileName = 'SMUDT'
BusinessOperation = 'HSDL'
FileDiscriminator = 'UserDefinedTable'
UserDefinedTable_UserTableCode = POSITION1
UserDefinedTable_LegislativeDataGroupName = POSITION2
UserDefinedTable_UserTableName = POSITION1
UserDefinedTable_RangeOrMatch = POSITION3
UserDefinedTable_UserRowTitle = POSITION4
UserDefinedTable_UserKeyUnits = POSITION5
RETURN
   BusinessOperation,FileDiscriminator,FileName,UserDefinedTable_UserTableCode,UserDefinedTable_LegislativeDataGroupName,UserDefinedTable_UserTableName,UserDefinedTable_RangeOrMatch,UserDefinedTable_UserRowTitle,UserDefinedTable_UserKeyUnits
}
ELSE
OUTPUTVALUE='NONE'
RETURN OUTPUTVALUE
/* End Formula Text */

Related Topics

- Guidelines for Using HCM Spreadsheet Data Loader
- How Data Is Uploaded Using HCM Spreadsheet Data Loader
- HCM Spreadsheet Data Loader Templates
- Create and Edit Spreadsheet Templates
- Guidelines for Designing Spreadsheet Templates
## 19 Loading Talent Objects

### Overview of Loading Goals, Goal Plans, and Goal Plan Sets

You can load Goal, Goal Plan, and Goal Plan Set objects using HCM Data Loader. Most of these objects have dependencies on other objects. This table identifies the other objects that must exist in the target environment. If you’re loading some or all of those objects with goals, goal plans, or goal plan sets, then follow the load order shown in the table.

<table>
<thead>
<tr>
<th>Object</th>
<th>Object Load Order</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal Plan</td>
<td>1. Worker and primary assignment</td>
<td>Review periods must exist in the target environment before you upload associated goal plans.</td>
</tr>
<tr>
<td></td>
<td>2. Eligibility profile</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Organization</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Goal plan document type</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5. Goal plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Organizations are required for organization goal plans.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A Goal Plan Document Types component is required when the IncludeInPerfDoc attribute is set to Y.</td>
</tr>
<tr>
<td>Goal Plan Set</td>
<td>1. Goal plan</td>
<td>Review periods must exist in the target environment before you upload associated goal plan sets.</td>
</tr>
<tr>
<td></td>
<td>2. Worker and primary assignment</td>
<td>Goal Plan Set objects can refer to goal plans.</td>
</tr>
<tr>
<td></td>
<td>3. Eligibility profile</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Goal plan set</td>
<td></td>
</tr>
<tr>
<td>Goal (library)</td>
<td>1. Content Item</td>
<td>Library goals are loaded as content items.</td>
</tr>
<tr>
<td>Goal (organization)</td>
<td>1. Worker (line manager)</td>
<td>For organization goals:</td>
</tr>
<tr>
<td></td>
<td>2. Organization</td>
<td>• The line manager is the organization owner.</td>
</tr>
<tr>
<td></td>
<td>3. Goal plan</td>
<td>• Goal plans are required.</td>
</tr>
<tr>
<td></td>
<td>4. Goal</td>
<td></td>
</tr>
<tr>
<td>Goal (template)</td>
<td>1. Worker (human resource specialist) and assignment</td>
<td>Goal plans are required.</td>
</tr>
<tr>
<td></td>
<td>2. Goal plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Goal</td>
<td></td>
</tr>
<tr>
<td>Goal (worker development)</td>
<td>1. Worker</td>
<td>Worker development goals aren’t associated with goal plans.</td>
</tr>
<tr>
<td></td>
<td>2. Content Item</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Goal</td>
<td></td>
</tr>
<tr>
<td>Goal (worker performance)</td>
<td>1. Worker and assignment</td>
<td>Worker performance goals must be associated with goal plans.</td>
</tr>
<tr>
<td></td>
<td>2. Goal plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Content Item</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Goal</td>
<td></td>
</tr>
</tbody>
</table>

### Related Topics
- Overview of Goal Management
Guidelines for Loading Goals

Goals represent the objectives of employees and organizations. This topic describes aspects of the Goal object that you must understand to load goals successfully.

Goal Types

This table identifies the types of goals that you can load using HCM Data Loader.

<table>
<thead>
<tr>
<th>Goal Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization goal</td>
<td>Created by a manager for a specific organization.</td>
</tr>
<tr>
<td>Template goal</td>
<td>Used in goal plans.</td>
</tr>
<tr>
<td>Worker goal</td>
<td>Assigned to a worker by a manager or human resource specialist. Two types of worker goals exist:</td>
</tr>
<tr>
<td></td>
<td>• Development goals</td>
</tr>
<tr>
<td></td>
<td>• Performance goals</td>
</tr>
</tbody>
</table>

**Note:** You can also load library goals using HCM Data Loader. You load library goals as profile `content items`. You can refer to library goals from other goal objects.

When you load goals in the Goal.dat file, you can include the `GoalTypeCode` attribute. This value is validated using the `HRG_GOAL_TYPE` lookup type. For `organization goals`, `GoalTypeCode` must be `PERFORMANCE`, which is the default value.

Goal Components

The components that you can supply when loading goals depend on the goal type. This table identifies when you can provide each component.

<table>
<thead>
<tr>
<th>Goal Component</th>
<th>Goal Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>All</td>
</tr>
<tr>
<td>Goal Action</td>
<td>All</td>
</tr>
</tbody>
</table>
| Goal Plan Goal       | • Worker `performance goals`
|                      | • Organization goals
|                      | • Template goals |
| Goal Target Outcome  | • Worker goals
|                      | • Template goals |
| Goal Measurement     | All             |

ORACLE
Goal Component | Goal Type
--- | ---

**Note:** You can specify only one Goal Measurement component for a development goal.

<table>
<thead>
<tr>
<th>Goal Access</th>
<th>• Worker goals</th>
</tr>
</thead>
</table>
| Goal Alignment | • Worker performance goals  
• Organization goals |

### Profile Options
This table identifies profile options that must be both enabled and set to Y for successful loading of goal components.

<table>
<thead>
<tr>
<th>Profile Option</th>
<th>What the Option Enables</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRG_ENABLE_GOAL_ALIGN</td>
<td>Alignment of goals in the Goal Alignment component</td>
</tr>
<tr>
<td>HRG_ENABLE_GRANT_ACCESS</td>
<td>Shared goals in the Goal Access component</td>
</tr>
<tr>
<td>HRG_ENABLE_OUTCOMES</td>
<td>Creation of goal target outcomes in the Goal Target Outcome component</td>
</tr>
<tr>
<td>HRG_ENABLE_TASK</td>
<td>Creation of goal tasks in the Goal Action component</td>
</tr>
</tbody>
</table>

- To enable profile options, use the **Manage Profile Options** task.
- To set the value of a profile option, use the **Manage Administrator Profile Values** task.

Perform both tasks in the Setup and Maintenance work area.

### Deleting Goals
You can delete Goal objects using HCM Data Loader. When you delete a parent Goal object, these child components are also deleted automatically:

- Goal Access  
- Goal Action  
- Goal Alignment, where **GoalId** or **AlignedGoalId** identifies the goal that you’re deleting  
- Goal Plan Goal  
- Goal Target Outcome

You can also delete child components individually, if required.

Goal Measurement components aren’t deleted automatically when you delete the parent goal using HCM Data Loader. You must delete Goal Measurement components explicitly before deleting the parent goal. This example Goal.dat file deletes a Goal Measurement component. The parent goal is identified by its source key.

```
METADATA|GoalMeasurement|SourceSystemId|SourceSystemOwner|GoalId(SourceSystemId)  
DELETE|GoalMeasurement|GoalMeasurement_001_SSID|VISION|WorkerGoal_001_SSID
```
This example Goal.dat file deletes a goal and its child components, other than Goal Measurement. The goal is identified by its source key.

```
METADATA|Goal|SourceSystemId|SourceSystemOwner
DELETE|Goal|WorkerGoal_001_SSID|VISION
```

### Examples of Loading Goals

This topic provides examples showing how to load worker goals and organization goals using HCM Data Loader.

### Creating a Worker Goal

This example Goal.dat file creates an active performance goal for a worker. In this file:

- The user-key attributes `WorkerPersonNumber` and `WorkerAssignmentNumber` identify the worker and the relevant assignment.
- The user-key attribute `AssignedByPersonNumber` identifies the human resource specialist or line manager who’s creating the goal.
- A Goal Plan Goal component must be provided.

> **Note:** For development goals, a Goal Plan Goal component isn’t required.

```
METADATA|Goal|SourceSystemId|SourceSystemOwner|GoalName|Description|StartDate|TargetCompletionDate|
WorkerPersonNumber|WorkerAssignmentNumber|StatusCode|GoalSourceCode|AllowKeyAttrUpdateFlag|
IncludeInPerfdocFlag|AssignedByPersonNumber
MERGE|Goal|WorkerGoal_001_SSID|VISION|WorkerGoal_001|WorkerGoal_001_Desc|2015/01/01|2015/12/31|8153787|EEEE8153787|Not started|HR|Y|Y|8153756
METADATA|GoalPlanGoal|SourceSystemId|SourceSystemOwner|GoalId(SourceSystemId)|GoalPlanName|GoalPlanStartDate|GoalPlanEndDate|PriorityCode
MERGE|GoalPlanGoal|GoalPlanGoal_001_SSID|VISION|WorkerGoal_001_SSID|GP_009_01|2015/01/01|2015/12/31|Medium
```

### Creating a Worker Goal Against a Library Goal

This example Goal.dat file creates an active performance goal against a library goal for a worker. HCM Data Loader doesn’t take values from the library goal. Instead, you must supply these values when you load the worker goal. The `ReferenceContentItemName` attribute is the user key for the library goal, which must exist.

```
METADATA|Goal|SourceSystemId|SourceSystemOwner|GoalName|Description|StartDate|TargetCompletionDate|
WorkerPersonNumber|WorkerAssignmentNumber|StatusCode|GoalSourceCode|AllowKeyAttrUpdateFlag|
IncludeInPerfdocFlag|AssignedByPersonNumber|ReferenceContentItemName
MERGE|Goal|WorkerGoal_002_SSID|VISION|WorkerGoal_002|WorkerGoal_002_Desc|2015/01/01|2015/12/31|8153787|EEEE8153787|Not started|HR|Y|Y|8153756|ZHRG-Completing Learning Path
```
Creating an Organization Goal

This example Goal.dat file creates and publishes an organization goal for a line manager. In this file:

- The user-key attribute **AssignedByPersonNumber** identifies the line manager who owns the organization goal. To publish the organization goal to the organization, set the **PublishedFlag** attribute to **Y**.

- The **GoalSourceCode** attribute is set to **Organization owner**. This value is required for organization goals.

- A Goal Plan Goal component must be provided.

```
METADATA|Goal|SourceSystemId|SourceSystemOwner|GoalName|Description|StartDate|TargetCompletionDate|
OrganizationName|StatusCode|GoalSourceCode|AllowKeyAttrUpdateFlag|AssignedByPersonNumber|PublishedFlag
MERGE|Goal|OrganizationGoal_001_SSID|VISION|OrganizationGoal_001|OrganizationGoal_001_Desc|2015/01/01|2015/12/31|Vision Corporation Enterprise|Not started|Organization owner|Y|8153786|Y

MERGE|GoalPlanGoal|GoalPlanGoal_002_SSID|VISION|OrganizationGoal_001_SSID |GP_009_01|2015/01/01|2015/12/31|Medium
```

Creating a Library Goal

Here's how to create library goals that can be referenced by other goal objects. You use ContentItem.dat, as in the following example, to add library goals.

```
METADATA|ContentItem|Name|ContextName|DateFrom|ItemDescription|ItemDate1|ItemDate2|ItemText1|ItemText2|ItemText3|ItemText4|ItemText5|ItemText6|ItemText7|ItemText8|ItemText9|ItemText10|ItemText11|ItemText12|ItemText13|ItemText14|ItemNumber1|ItemNumber2|ItemNumber3|ItemNumber4|ItemNumber5|ItemNumber6|ItemNumber7|ItemNumber8|ItemTextTl12|ItemTextTl11|ItemTextTl14|SourceSystemId|SourceSystemOwner
MERGE|ContentItem|HL Goal Library 2016|GOAL|2016/04/01|HL Goal Library 2016 Item Description|2016/04/01|2016/04/30|A|LOW|STRETCH|QUANTITATIVE|MIN|PERCENT|SHORT_TERM|DEVELOPMENT|ALL|FINANCE|HLGoal2016|10|10|20|50|50|50|ZHRA-TST-Measurement|Comment for the goal category|Testing Goal Library HDL|SSIDGL2016HL001|HCMQA-001

MERGE|ContentItem|HL Goal Library 2017|GOAL|2017/04/01|HL Goal Library 2017 Item Description|2017/04/01|2017/04/30|A|LOW|STRETCH|QUANTITATIVE|MIN|PERCENT|SHORT_TERM|DEVELOPMENT|ALL|FINANCE|HLGoal2017|10|10|20|50|50|50|ZHRA-TST-Measurement|Comment for the goal category|Testing Goal Library HDL|SSIDGL2017HL001|HCMQA-001

MERGE|ContentItem|HL Goal Library 2018|GOAL|2018/04/01|HL Goal Library 2018 Item Description|2018/04/01|2018/04/30|A|LOW|STRETCH|QUANTITATIVE|MIN|PERCENT|SHORT_TERM|DEVELOPMENT|ALL|FINANCE|HLGoal2018|10|10|20|50|50|50|ZHRA-TST-Measurement|Comment for the goal category|Testing Goal Library HDL|SSIDGL2018HL001|HCMQA-001

MERGE|ContentItem|HL Goal Library 2019|GOAL|2019/04/01|HL Goal Library 2019 Item Description|2019/04/01|2019/04/30|A|LOW|STRETCH|QUANTITATIVE|MIN|PERCENT|SHORT_TERM|DEVELOPMENT|ALL|FINANCE|HLGoal2019|10|10|20|50|50|50|ZHRA-TST-Measurement|Comment for the goal category|Testing Goal Library HDL|SSIDGL2019HL001|HCMQA-001
```

After you import the .zip file containing the content items, verify you can add the new goal libraries to a user by navigating to **Me > Career and Performance > Career Development**. In the **Development Plan: Active Goals** section, click the **Add Goal** menu and select **Library Goal**. Confirm that the recently imported library goals appear in the selection list.
Guidelines for Loading Goal Plans

Goal plans are used for the mass assignment of goals to worker populations. This topic describes aspects of the Goal Plan object that you must understand to load goal plans successfully.

Assigning Goals to Workers
When you load a Goal Plan object, the plan isn’t automatically assigned to workers. You can assign the goal plan to workers in either of these ways:

- Schedule goal plan assignments using the Manage Goal Plans task.
- Load the Goal Plan Assignment component of the goal plan.

The Mass Request Component
The Mass Request component identifies who’s requesting the goal plan. You can identify the plan requester on the goal plan itself by supplying the RequestSubmissionDate and ReqSubmittedByPersonNumber attributes, for example. In this case, the Mass Request component is created automatically and you don’t have to supply it. However, you must supply a Mass Request component when you create a goal plan:

- If you don’t identify the plan requester on the Goal Plan object.
- To associate a source key with the Mass Request record.
  Alternatively, you can update the default source key later using the Source Key object.
- If you’re loading child components of the Mass Request component.
  The child components of the Mass Request component identify the employee population to which the goal plan is to be assigned.

Deleting Goal Plans
You can delete Goal Plan objects using HCM Data Loader. However, don’t delete goal plans that have been assigned to workers. When you delete a parent Goal Plan object, these child components are also deleted automatically:

- Goal Plan Document Types
- Mass Request

Note: When you delete a Mass Request component, its child components are deleted automatically.

You can also delete child components individually, if required.

Goal Plan Goal and Goal Plan Assignment components aren’t deleted automatically when you delete the parent Goal Plan component using HCM Data Loader. You must delete those components explicitly before you delete the parent goal plan. This example GoalPlan.dat file deletes the Goal Plan Goal component that links goal plan GP_001_01 with template goal TemplateGoal_001.

```plaintext
METADATA|GoalPlanGoal|SourceSystemId|SourceSystemOwner|GoalPlanExternalId
DELETE|GoalPlanGoal|GPG_001_01_SSID|VISION|GP_001_01
```
This example Goal.dat file deletes the template goal TemplateGoal_001 (SourceSystemId: TemplateGoal_001_SSID).

```plaintext
METADATA|Goal|SourceSystemId|SourceSystemOwner
DELETE|Goal|TemplateGoal_001_SSID|VISION
```

This example GoalPlan.dat file deletes the goal plan GP_001_01 and relevant child records.

```plaintext
METADATA|GoalPlan|GoalPlanExternalId
DELETE|GoalPlan|GP_001_01
```

**Related Topics**

- Goal Plans

**Examples of Loading Goal Plans**

This topic provides examples showing how to load Goal Plan objects using HCM Data Loader.

### Creating a Goal Plan

This example GoalPlan.dat file loads a *goal plan* without a Mass Request component. In this case, a mass request record for the goal plan is created automatically using the values passed for the `ReqSubmittedByPersonNumber` and `RequestSubmissionDate` attributes.

```plaintext
METADATA|GoalPlan|GoalPlanExternalId|GoalPlanName|GoalPlanTypeCode|GoalPlanActiveFlag|EnableWeightingFlag|StartDate|EndDate|EnforceGoalWeightFlag|GoalAccessLevelCode|IncludeInPerfdocFlag|ReqSubmittedByPersonNumber|RequestSubmissionDate|ReviewPeriodName
MERGE|GoalPlan|GP_001_01|GP_001_01|ORA_HRG_WORKER|A|Y|2015/01/01|2015/12/31|Y|HR specialist and manager|N|8153756|2015/09/20|Default Review Period - Vision Corporation Enterprise
```

### Creating a Goal Plan with Document Types

This example GoalPlan.dat file creates a goal plan with the `IncludeInPerfdocFlag` attribute set to `Y` and assigns a document type.

```plaintext
METADATA|GoalPlan|GoalPlanExternalId|GoalPlanName|GoalPlanTypeCode|GoalPlanActiveFlag|EnableWeightingFlag|StartDate|EndDate|EnforceGoalWeightFlag|GoalAccessLevelCode|IncludeInPerfdocFlag|ReqSubmittedByPersonNumber|RequestSubmissionDate|ReviewPeriodName
MERGE|GoalPlan|GP_001_02|GP_001_02|ORA_HRG_WORKER|A|Y|2016/01/01|2016/12/31|Y|HR specialist and manager|Y|8153756|2016/10/04|Default Review Period - Vision Corporation Enterprise
MERGET|GoalPlanDocTypes|GoalPlanExternalId|DocTypeName
MERGE|GoalPlanDocTypes|GP_001_02|Annual Evaluations
```

### Creating a Goal Plan with Eligibility Criteria

This example GoalPlan.dat file creates a goal plan with Mass Request and Eligibility Profile Object components.

```plaintext
METADATA|GoalPlan|GoalPlanExternalId|GoalPlanName|GoalPlanTypeCode|GoalPlanActiveFlag|EnableWeightingFlag|StartDate|EndDate|EnforceGoalWeightFlag|GoalAccessLevelCode|IncludeInPerfdocFlag|ReviewPeriodName
```
Guidelines for Loading Goal Plan Sets

Goal plan sets are used for the mass assignment of related goal plans to worker populations. This topic describes aspects of the Goal Plan Set object that you must understand to load goal plan sets successfully.

Assigning Goal Plan Sets to Workers

When you load a goal plan set, it’s not automatically assigned to workers. You can assign the goal plan set to workers in either of these ways:

- Schedule goal plan set assignments using the Manage Goal Plan Sets task.
- Load the Goal Plan Assignment component of the Goal Plan object. You may use this approach, for example, to assign goals following the transfer of workers from one legal entity to another.

The Mass Request Component

The Mass Request component identifies who’s requesting the goal plan set. You can identify the plan requester on the goal plan set itself by supplying the RequestSubmissionDate and ReqSubmittedByPersonNumber attributes, for example. In this case, the Mass Request component is created automatically and you don’t have to supply it. However, you must supply a Mass Request component when you create a goal plan set:

- If you don’t identify the plan requester on the Goal Plan Set object.
- To associate a source key with the mass request record.
  Alternatively, you can update the default source key later using the Source Key object.
- If you’re loading child components of the Mass Request component.
  The child components of the Mass Request component identify the employee population to which the goal plan set is to be assigned.

Deleting Goal Plan Sets

You can delete Goal Plan Set objects using HCM Data Loader. When you delete a parent goal plan set, these child components are also deleted automatically:

- Goal Plan Set Plan
- Mass Request

⚠️ Note: When you delete a Mass Request component, its child components are deleted automatically.

Don’t delete goal plan sets that have already been assigned to workers.
This example GoalPlanSet.dat file deletes a goal plan set and its child components.

```
METADATA|GoalPlanSet|GoalPlanSetExternalId
DELETE|GoalPlanSet|GPS_006_01
```

**Related Topics**

- Goal Plan Sets

**Examples of Loading Goal Plan Sets**

This topic provides examples showing how to load Goal Plan Set objects using HCM Data Loader.

**Creating a Goal Plan Set with Goal Plans**

This example GoalPlanSet.dat file creates a goal plan set and includes two goal plans.

```
METADATA|GoalPlanSet|GoalPlanSetExternalId|GoalPlanSetName|Description|GoalPlanSetActiveFlag|StartDate|EndDate|ReviewPeriodName
MERGE|GoalPlanSet|GPS_002_01|GPS_002_01|GPS_002_01|A|2015/01/01|2015/12/31|Default Review Period - Vision Corporation Enterprise
COMMENT Assign two goal plans without worker or eligibility profile assignment to the goal plan set
METADATA|GoalPlanSetPlan|GoalPlanSetExternalId|GoalPlanExternalId|Weighting
MERGE|GoalPlanSetPlan|GPS_002_01|GP_001_02|60
MERGE|GoalPlanSetPlan|GPS_002_01|GP_001_03|40
```

In this example, the goal plan set has no Mass Request component. Therefore, the mass request record for the goal plan set is created automatically using the `ReqSubmittedByPersonNumber` and `RequestSubmissionDate` attribute values.

**Creating a Goal Plan Set with a Mass Request Component**

This example GoalPlanSet.dat file creates a goal plan set with a Mass Request component. You don’t provide the `ReqSubmittedByPersonNumber` and `RequestSubmissionDate` attributes for the Goal Plan Set component in this case.

```
METADATA|GoalPlanSet|GoalPlanSetExternalId|GoalPlanSetName|Description|GoalPlanSetActiveFlag|StartDate|EndDate|ReviewPeriodName
MERGE|GoalPlanSet|GPS_001_02|GPS_001_02|GPS_001_02|A|2016/01/01|2016/12/31|Default Review Period - Vision Corporation Enterprise
```

```
METADATA|MassRequest|GoalPlanSetExternalId|ReqSubmittedByPersonNumber
MERGE|MassRequest|GPS_001_02|8153756
```

**Creating a Goal Plan Set with Mass Request and Mass Request Hierarchy Components**

This example GoalPlanSet.dat file creates a goal plan set with Mass Request and Mass Request Hierarchy components.

```
METADATA|GoalPlanSet|GoalPlanSetExternalId|GoalPlanSetName|Description|GoalPlanSetActiveFlag|StartDate|EndDate|ReviewPeriodName
```

```
METADATA|GoalPlanSet|GoalPlanSetExternalId|GoalPlanSetName|Description|GoalPlanSetActiveFlag|StartDate|EndDate|ReviewPeriodName
```
Examples of Loading Mass Actions for Performance Documents

Using HCM Data Loader, you can create performance documents in bulk. This topic provides examples showing how to load Mass Actions for Performance Documents objects with Attachment and Participants components. In these examples, all components are referenced by their user keys.

Loading Mass Actions for Performance Documents

This example PerfDocComplete.dat file creates a performance document for a specified worker assignment.

METADATA|PerfDocComplete|AssignmentNumber|CustomaryName|StartDate|EndDate|Operation|ManagerAssignmentNumber
MERGE|PerfDocComplete|EEEE100000017121468|zhra-kudosintegration2018|2018/01/01|2018/12/31|ORA_CREATE_PD|EEEE8153756

The Operation attribute identifies the action to be performed. The following table lists valid operation values, indicates whether they’re defined in the Performance Administration Action lookup (ORA_HRA_ADMIN_ACTION), and provides information about any dependencies.

<table>
<thead>
<tr>
<th>Operation Value</th>
<th>Defined in Lookup</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA_CREATE_PD</td>
<td>Yes</td>
<td>Creates a performance document. The performance template period and worker assignment must exist.</td>
</tr>
<tr>
<td>ORA_COMPLETE_PD</td>
<td>Yes</td>
<td>Completes a performance document. The performance document must exist.</td>
</tr>
<tr>
<td>ORA_DELETE_PD</td>
<td>Yes</td>
<td>Deletes a performance document. The performance document must exist and its status must be Canceled.</td>
</tr>
<tr>
<td>ORA_ADD_PARTICIPANT</td>
<td>No</td>
<td>Use this value for all supported actions on performance document participants.</td>
</tr>
</tbody>
</table>

Note: If you need to mass update participants or attachments, you must delete and load them again via HCM Data Loader.
Loading Attachments

This example PerfDocComplete.dat file loads an attachment to an existing performance document.

```
METADATA|PerfDocComplete|AssignmentNumber|CustomaryName|Operation
MERGE|PerfDocComplete|EEEE8153772|zhra-kudosintegration2018|ORA_ATTACH_PD
METADATA|Attachment|CustomaryName|AssignmentNumber|Title|DataTypeCode|URLorTextorFileName|MimeType|File|Description
MERGE|Attachment|zhra-kudosintegration2018|EEEE8153772|JBlum Resume|FILE|jblum.txt|text/plain|jblum.txt|Jorge Blum Resume
```

These rules apply when you load attachments:

- You must include the parent performance document in the PerfDocComplete.dat file. Set the **Operation** attribute to **ORA_ATTACH_PD** for both MERGE and DELETE instructions for the Attachment component.
- The **MimeType** value is the Multipurpose Internet Mail Extensions (MIME) value, as defined and standardized by the Internet Engineering Task Force in RFC 6838.
- The attachment itself is a binary large object, referenced by the **File** attribute. You include the attachment in the BlobFiles folder in the .zip file that you upload.
- You can only bulk delete attachments and load them again. Update action isn’t supported.

Loading Participants

This example PerfDocComplete.dat file deletes a participant record from an existing performance document.

```
METADATA|PerfDocComplete|AssignmentNumber|CustomaryName|Operation
MERGE|PerfDocComplete|EEEE8153772|FY2018 Annual Performance Evaluation with Participant Feedback|ORA_ADD_PARTICIPANT
METADATA|Participant|AssignmentNumber|CustomaryName|ParticipantRoleTypeCode|ParticipantPersonNumber
DELETE|Participant|EEEE8153772|FY2018 Annual Performance Evaluation with Participant Feedback|Colleague|ZHRA-8154961
```

These rules apply when you load participants:

- When you create participants, the parent performance document must already exist. You can’t both create a performance document and add a participant in the same PerfDocComplete.dat file.
- You must include the parent performance document in the PerfDocComplete.dat file. Set the **Operation** attribute to **ORA_ADD_PARTICIPANT** for both MERGE and DELETE instructions for the Participants component.
- The participant person record, as identified by the **ParticipantPersonNumber** attribute value, must exist.
- The **ParticipantRoleTypeCode** value must have been selected in the performance template.
- You can only bulk delete participants and load them again. Update action isn’t supported.
Examples of Loading Succession Plans

A succession plan identifies a group of workers as candidates to replace key personnel in an organization. The workers are those who are ready now, or can develop the necessary skills, for jobs and positions that aren't currently vacant. Succession plans include assessments of a worker's readiness to move into a vacancy and the risk and impact of the worker's loss. This topic provides some examples showing how to load succession plans.

Creating Succession Plans

This example SuccessionPlan.dat file creates an incumbent succession plan using source keys.

```
METADATA|SuccessionPlan|SourceSystemOwner|SourceSystemId|PlanName|PlanType|Status|AccessTypeCode|IncumbentPersonId(SourceSystemId)
MERGE|SuccessionPlan|VISION|SP816003|Name of the incumbent succession plan|INCUMBENT|ACTIVE|PRIVATE|VISION006
METADATA|SuccessionPlanOwner|SourceSystemOwner|SourceSystemId|PlanId(SourceSystemId)|OwnerTypeCode|PersonId(SourceSystemId)
MERGE|SuccessionPlanOwner|VISION|SPO816003|SP816003|ADMINISTRATOR|VISION013
METADATA|SuccessionPlanCandidate|SourceSystemOwner|SourceSystemId|PlanId(SourceSystemId)|Status|CandidateReadiness
MERGE|SuccessionPlanCandidate|VISION|SPC816003|SP816003|VISION014|ACTIVE|READY_NOW
```

This example SuccessionPlan.dat file creates a job succession plan. It uses user keys to reference the job to which the succession plan relates, the succession plan owner, and its candidates.

```
METADATA|SuccessionPlan|SourceSystemOwner|SourceSystemId|PlanName|PlanType|Status|AccessTypeCode|JobCode|JobSetCode
MERGE|SuccessionPlan|VISION|SP_MRKT_DIR|Name of the job succession plan|JOB|ACTIVE|PRIVATE|HSDL_MRKT_DIR|COMMON
METADATA|SuccessionPlanOwner|SourceSystemOwner|SourceSystemId|PlanId(SourceSystemId)|OwnerTypeCode|PlanOwnerPersonNumber
MERGE|SuccessionPlanOwner|VISION|SPO_MRKT_DIR|SP_MRKT_DIR|ADMINISTRATOR|1023853
METADATA|SuccessionPlanCandidate|SourceSystemOwner|SourceSystemId|PlanId(SourceSystemId)||PlanCandidatePersonNumber|Status|CandidateReadiness
MERGE|SuccessionPlanCandidate|VISION|SPC009_MRKT_DIR|SP_MRKT_DIR|1023119|ACTIVE|READY_NOW
MERGE|SuccessionPlanCandidate|VISION|SPC008_MRKT_DIR|SP_MRKT_DIR|1032314|ACTIVE|READY_NOW
MERGE|SuccessionPlanCandidate|VISION|SPC007_MRKT_DIR|SP_MRKT_DIR|1032112|ACTIVE|READY_NOW
```

These rules apply when you create a succession plan using HCM Data Loader:

- You must supply a Success Plan Owner component, and the OwnerTypeCode attribute must be ADMINISTRATOR. A succession plan must always have an administrator owner.
- Incumbents, plan owners, and plan candidates must have active assignments.
Examples of Loading Talent Pools

A talent pool is a group of workers. You use talent pools to manage the development of the pool members for specific purposes. For example, you can track potential candidates for a *succession plan* in a talent pool or create a *talent review* meeting based on a talent pool. This topic provides examples showing how to create and manage talent pools using HCM Data Loader.

Creating a Talent Pool

A talent pool can reference any combination of *job, job family, model profile, position*, department, and *grade*. These objects, which can define the purpose of the talent pool, have no effect on its processing. However, they must exist in the target environment before you load a talent pool that references them. This example TalentPool.dat file creates a talent pool for a job. Source keys are used to identify both the talent pool and the associated job.

```plaintext
METADATA|TalentPool|SourceSystemOwner|SourceSystemId|PoolName|PoolTypeCode|Description|
OwnerPersonId(SourceSystemId)|Status|JobId(SourceSystemId)
MERGE|TalentPool|VISION|TP_T1|Job Chief Exec|TALENT|Talent pool description|1008|A|93079
```

Always set the *PoolTypeCode* attribute to **TALENT**. You must include the *Status* attribute. If you supply no value, then the default status is **A** (Active). Set the status to **I** (Inactive) to inactivate the talent pool.

The Talent Pool component defines the talent pool and the initial *pool owner*. To identify additional pool owners, you use the Talent Pool Owner component. Additional pool owners are optional, but multiple owners are recommended. Talent pool owners can see talent pools in the application. Use the Talent Pool Member component to define the pool members. This example TalentPool.dat file creates a talent pool with pool members. Source keys are used to identify the talent pool and reference foreign objects.

```plaintext
METADATA|TalentPool|SourceSystemOwner|SourceSystemId|PoolName|PoolTypeCode|Description|
OwnerPersonId(SourceSystemId)|Status
MERGE|TalentPool|VISION|TP_T2|Member Talent Pool|TALENT|Talent pool description|1011|A
METADATA|PoolMember|SourceSystemOwner|SourceSystemId|PoolId(SourceSystemId)|MemberId(SourceSystemId)
MERGE|PoolMember|VISION|TP_PM_T2_1012|TP_T2|1012
MERGE|PoolMember|VISION|TP_PM_T2_1013|TP_T2|1013
```

Translating a Talent Pool Name

This example TalentPoolTranslation.dat file translates the name of an existing talent pool. It identifies the existing record by its source key.

```plaintext
METADATA|TalentPoolTranslation|SourceSystemOwner|SourceSystemId|PoolName|Language
MERGE|TalentPoolTranslation|VISION|TP_T1|Spanish Translated Pool Name|ES
MERGE|TalentPoolTranslation|VISION|TP_T1|French Translated Pool Name|FR
```

This example TalentPoolTranslation.dat file translates the name of an existing talent pool. It identifies the existing record by its user key.

```plaintext
METADATA|TalentPoolTranslation|BasePoolName|PoolStatus|PoolName|Language
MERGE|TalentPoolTranslation|Job Chief Exec|A|Spanish Translated Pool Name|ES
MERGE|TalentPoolTranslation|Job Chief Exec|A|French Translated Pool Name|FR
```
Deleting Talent Pools

You can delete a talent pool using HCM Data Loader. You can also:

- Make the talent pool inactive.
- Delete talent pool owners, provided that at least one pool owner remains.
- Delete Talent Pool Member components.

This example TalentPool.dat file deletes an existing Talent Pool Owner component. It identifies the pool owner record by its source key.

| METADATA|PoolOwner|SourceSystemOwner|SourceSystemId
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE</td>
<td>PoolOwner</td>
<td>VISION</td>
<td>TP_PO_T2_3423</td>
</tr>
</tbody>
</table>

This example deletes the Talent Pool, itself. When you delete this parent object, you also delete its children, such as Talent Pool Owner.

| METADATA|TalentPool|PoolName|PoolTypeCode|Status
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE</td>
<td>TalentPool</td>
<td>Test KV 3</td>
<td>TALENT</td>
<td>A</td>
</tr>
</tbody>
</table>

Related Topics

- Talent Pools

Guidelines for Loading Talent Profile Data

The talent profile objects are Talent Profile, Content Item, Content Items Relationship, Education Establishment, and Rating Model. This topic describes aspects of the talent profile objects that you must understand to load them successfully using HCM Data Loader.

Loading Talent Profiles

Talent profiles include person profiles and model profiles. A person profile contains information about a worker’s skills, qualifications, accomplishments, and career preferences. A model profile contains information about the skills and qualifications required for a job or position. You can load both person and model profiles using HCM Data Loader.

The following table summarizes key information about some components of the Talent Profile object.

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile Item</td>
<td>Holds the data for a content item. Content items are skills, qualifications, and qualities belonging to content types in the content library. For example, Communication Skills may be a content item in the Competencies content type. Profile Item components must be unique in the profile.</td>
</tr>
<tr>
<td></td>
<td><strong>Tip:</strong> Duplicate profile items have the same qualifiers and overlapping start and end dates.</td>
</tr>
<tr>
<td>Profile Keyword</td>
<td>Holds keywords for areas of expertise and areas of interest in person profiles.</td>
</tr>
<tr>
<td></td>
<td>Not valid for model profiles.</td>
</tr>
<tr>
<td>Component</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Profile Relation</td>
<td>Links a model profile to a job or position. These rules apply:</td>
</tr>
<tr>
<td></td>
<td>• You can’t link a model profile to a job family or organization when you upload the profile using HCM Data Loader. Any model profile that you load for those structures is therefore for information purposes only.</td>
</tr>
<tr>
<td></td>
<td>• You can’t associate a job or position more than once with the same profile.</td>
</tr>
<tr>
<td></td>
<td>Not valid for person profiles.</td>
</tr>
</tbody>
</table>

A person can have only one person profile. You must supply a reference to the person for whom you’re creating the person profile. You must not supply a person reference for a model profile.

**Note:** The Description attribute of the Talent Profile object isn’t required. However, you’re recommended always to provide it for a person profile and include the worker name. Otherwise, the person profile doesn’t appear in search results when you perform a basic search on the Search: Profiles page.

### Loading Content Items

A content item is a specific occurrence of a content type. The associated content types and their field properties must exist in the target environment before you load content items to the content library. The field properties of the content type represent the information that you can capture for the content item.

### Loading Content Items Relationships

A content items relationship defines the relationship between two content items. A content item can be either the parent or the child of another content item. These rules apply:

- The two content items must belong to different content types.
- You can’t create more than one relationship between the same two content items. For example, content item A can’t be both the parent and the child of content item B.
- You can’t create a relationship between a content item and itself.
- The start and end dates of the content items relationship must not be outside the date ranges of the associated content items.
- The relationship between the associated content types must exist in the target environment before you load the related content items.

### Loading Education Establishments

An education establishment is a school, college, university, or other organization that workers select when they add education information to their profiles. You may obtain a list of education establishments from a third party, for example. You can upload this information using HCM Data Loader.

These geography structures must exist in the target environment before you load education establishments that refer to them:

- Country
- State or Province
Loading Rating Models

You use rating models to rate workers on their performance and proficiency in the skills and qualities in the person profile. You can also use rating models to specify target proficiency levels for items in a model profile. You must include the parent Rating Model component in the same file as any child Rating Level and Rating Category components that you're loading.

Related Topics

- Oracle Fusion Profile Management Components: How They Work Together

Examples of Loading Talent Review Meetings

Talent review meetings are the main component of the talent review process, which evaluates trends, assesses strengths, and addresses areas of risk for the organization. Participants in the talent review meeting review worker profile, performance, goals, and compensation data. This topic shows how to create a talent review meeting using HCM Data Loader and how to update the facilitator for an existing meeting.

Creating a Talent Review Meeting

This example TalentReviewMeeting.dat file creates a Talent Review Meeting object. The meeting has Facilitator, Participant, Reviewee, Review Content, Talent Pool, and Succession Plan components.

> Note: Source keys identify each record. You must use source keys if you plan to update the facilitator, participant, or reviewee records later. The user key for these records includes the person number. Therefore, you can't update the person number to refer to a different person if you're identifying the record by the user key alone. You're recommended always to use source keys when creating talent review meetings so that you can update the meetings as necessary.
These rules apply to Talent Review Meeting objects that you create using HCM Data Loader:

- The **DashboardTemplateName** attribute identifies the configuration template used for the meeting. The template is created in the Setup and Maintenance work area using the **Configure Talent Review Dashboard Options** task. The referenced template must exist before you load the talent review meeting record.

- Any facilitator, participant, reviewee, **talent pool**, and **succession plan** records that are referenced by the talent review meeting record must already exist. Alternatively, you can include them in the same .zip file.

- You must identify at least one facilitator.

- The **MeetingStatusCode** attribute must be set to **NOT_STARTED** and the **MeetingSubmitStatusCode** attribute can be set to either **DRAFT** or **SCHEDULED** for new talent review meetings.

### Updating a Talent Review Meeting

The following table lists actions that you can perform when updating a talent review meeting using HCM Data Loader. Permitted actions depend on the meeting status.

<table>
<thead>
<tr>
<th>Meeting Status</th>
<th>Permitted Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not started</td>
<td>• Change the meeting template.</td>
</tr>
<tr>
<td></td>
<td>• Change the meeting status to <strong>In progress</strong> or <strong>Canceled</strong>.</td>
</tr>
<tr>
<td></td>
<td>• Add or remove reviewees, participants, and facilitators.</td>
</tr>
<tr>
<td>In progress</td>
<td>• Change the meeting status to <strong>Not started</strong> or <strong>Canceled</strong>.</td>
</tr>
<tr>
<td></td>
<td>• Add or remove reviewees, participants, and facilitators.</td>
</tr>
<tr>
<td>Complete</td>
<td>Update the facilitator.</td>
</tr>
<tr>
<td>Canceled</td>
<td>Change the meeting status to <strong>Not started</strong>.</td>
</tr>
<tr>
<td>Process job in progress</td>
<td>None.</td>
</tr>
</tbody>
</table>
Updating the Facilitator

This example TalentReviewMeeting.dat file updates the facilitator for an existing meeting. This example uses source keys. You can’t use user keys alone to update the facilitator, as the FacilitatorPersonNumber attribute that identifies the new facilitator also identifies the record to update.

```
METADATA|Facilitator|SourceSystemOwner|SourceSystemId|MeetingId(SourceSystemId) |
FacilitatorPersonId(SourceSystemId)
MERGE|Facilitator|VISION|MrktMgr071018_Fac1|MrktMgr071018|HDL014
```

Tip: If you’re making multiple changes to the same talent review meeting, then include the Talent Review Meeting record in the same .dat file to ensure that all changes are processed together.

Related Topics

- Talent Review
- Talent Review Template

FAQs for Loading Talent Objects

Can I load any type of requisition using the Requisition object?

No. You can load:

- Vacancies, which are placeholders for future headcount. Use vacancies if you don’t have a dedicated recruiting application.
- Oracle Talent Acquisition Cloud requisitions.

You can’t load Oracle Recruiting Cloud requisitions using the Requisition object. To load Oracle Recruiting Cloud requisitions, use the Job Requisition object.
20 Loading Learning Objects

Examples of Loading Classroom Resources

A classroom resource is the location of an on-site, instructor-led activity. This topic shows how to load Classroom Resource objects using HCM Data Loader.

Creating Classroom Resources

This example ClassroomResource.dat file shows how to create a Classroom Resource object using source keys.

```
COMMENT Create a Classroom Resource object using source keys
METADATA|ClassroomResource|SourceSystemOwner|SourceSystemId|ClassroomResourceNumber|Title|Description|Capacity|ContactId(SourceSystemId)|LocationId(SourceSystemId)|OwnedByPersonId(SourceSystemId)
MERGE|ClassroomResource|VISION|201803081352|RSC-201803081352-HDL|Title:RSC 201803081352 HDL|Desc:RSC 201803081352 HDL|30|300213211235|StLouisMO|300213211235
```

This example ClassroomResource.dat file shows how to create a Classroom Resource object using user keys.

```
COMMENT Create a Classroom Resource object using user keys
METADATA|ClassroomResource|ClassroomResourceNumber|Title|Description|Capacity|ContactNumber|LocationCode|SetCode|OwnedByPersonNumber
MERGE|ClassroomResource|RSC-201803081355-HDL|Title:RSC 201803081355 HDL|Desc:RSC 201803081355 HDL|30|8153757|DDT_US_Allenton _St Louis_MO|COMMON|8153757
```

Examples of Loading Course Offering Pricing Defaults

Course Offering Pricing Defaults objects are the default pricing-rule values for a course offering. This topic shows how to load Course Offering Pricing Defaults objects using HCM Data Loader.

Creating Course Offering Pricing Defaults

This example CourseOfferingPricingDefaults.dat file identifies the Course Offering Pricing Defaults object by its user key.

```
METADATA|CourseOfferingPricingDefaults|EffectiveStartDate|PricingRuleNumber|CurrencyCode|OfferingType|CourseNumber
MERGE|CourseOfferingPricingDefaults|2018/02/04|PR-201802041720-HDL|USD|ORA_SP|COURSE-201802041711-HDL
```

Creating Course Offering Pricing Components

You can create one or more Course Offering Pricing Components for a Course Offering Pricing Defaults object. This example CourseOfferingPricingDefaults.dat file creates two Course Offering Pricing Components, which are identified by their user keys.
Guidelines for Loading Courses

A course is teaching on a single subject with specific outcomes. A course has offerings that the learner can take to achieve the course outcomes. For example, a course may have one offering that’s instructor led and another that’s self-paced with an associated video. A Course object provides administrative information about the course. It includes one or more Offering components, each of which provides administrative information about a single offering. This topic describes aspects of the Course object that you must understand to load courses successfully using HCM Data Loader.

Offering Components

These rules apply to Offering components that you load using HCM Data Loader:

- Valid values of the Offering Type attribute are defined in the ORA_WLF_LEARN_DELIVERY_MODE lookup type. Supply one of these values:

<table>
<thead>
<tr>
<th>Offering Type Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA_BLENDED</td>
<td>Blended</td>
</tr>
<tr>
<td>ORA_SP</td>
<td>Self-paced</td>
</tr>
<tr>
<td>ORA_ILT</td>
<td>Instructor-led</td>
</tr>
</tbody>
</table>

You can’t update the offering type of an existing offering.

- The Offering component references an evaluation using the Questionnaire object named on the QuestionnaireCode attribute. To identify the default evaluation, set the attribute to -1.

Instructor Led Activity Components

You must supply a value for either the VirtualClassroomURL or the ClassroomResourceNumber attribute. If you supply:

- A VirtualClassroomURL value, then the current activity is an online activity
- A ClassroomResourceNumber value, then the current activity is an on-site activity
- Both VirtualClassroomURL and ClassroomResourceNumber values, then the current activity is an on-site activity

The Instructor Led Activity component has child components that identify resources required for the activity.

- You can identify a single classroom resource on the Instructor Led Activity component itself. To identify additional classroom resources, use the Classroom Reservation component. The Classroom Resource object referenced from the Classroom Reservation component must exist.
To add instructors, use the Instructor Reservation component. The Instructor Resource object referenced from the Instructor Reservation component must exist.

Self-Paced Activity Components
If you supply a value for the ContentNumber attribute, then the activity is classified as a self-paced, online activity. Otherwise, the activity is classified as a self-paced, offline activity.

Related Topics
• Programmatic Loading of Rich Media Content (2374277.1)

Examples of Loading Courses
This topic provides examples showing how to load Course objects and their components using HCM Data Loader.

Loading Course Objects
This example Course.dat file shows how to load a Course object using user keys.

```
METADATA|Course|EffectiveStartDate|CourseNumber|Title|ShortDescription|PublishStartDate|PublishEndDate|
MinimumExpectedEffort|MaximumExpectedEffort|CurrencyCode|MinimumPrice|MaximumPrice|TrailerLiNumber|
OwnedByPersonNumber|SourceType|SourceId|SourceInfo
MERGE|Course|2018/02/04|COURSE-201802041711-HDL|Title:COURSE 201802041711|ShortDesc:COURSE201802041711|
2018/02/04|2018/02/04|0|1|USD|0|0|OLC103778|8153757|FUSION_HCM_LEARNING|201802041711|HCM Learning Source Info for 201802041711 HDL Update
```

Loading Offering Components
This example Course.dat file shows how to load Offering components using user keys.

```
METADATA|Offering|EffectiveStartDate|OfferingNumber|Title|OfferingType|PublishStartDate|PublishEndDate|
LanguageCode|MinimumCapacity|MaximumCapacity|QuestionnaireCode|QstnrRequiredForCompletion|CourseNumber|
OwnedByPersonNumber|SourceType|SourceId|SourceInfo
MERGE|Offering|2018/02/04|OFFERING-201802041712-HDL|Title:Offering 201802041712|ORA_BLENDED|2018/02/04|
2018/02/04|2018/02/04|0|1|USD|0|0|COURSE-201802041711-HDL|8153757|FUSION_HCM_LEARNING|201802041712|HCM Learning Source Info for 201802041712
```

Loading Instructor Led Activity Components
This example Course.dat file shows how to load Instructor Led Activity components using user keys. It shows how to load child components of the Instructor Led Activity, such as Classroom Reservation and Instructor Reservation, to support the activity.

```
METADATA|InstructorLedActivity|EffectiveStartDate|ActivityNumber|Title|ActivityDate|ActivityStart|ActivityEnd|TimeZone|ExpectedEffort|SelfCompleteFlag|OfferingNumber|VirtualClassroom|ClassroomResourceNumber|SourceType|SourceId|SourceInfo
```

Oracle Human Capital Management Cloud
Integrating with HCM

Chapter 20
Loading Learning Objects

313
Loading Learning Objects

Chapter 20

Loading Self-Paced Activity Components

This example Course.dat file shows how to load Self-Paced Activity components using user keys.

Examples of Associating Global Access Groups

You can create Global Access Groups to classify groups of students by a common characteristic, such as their major. Then you can assign the Global Access Group group to one or more learning items within the catalog. Students can view the learning items for their Global Access Groups. This topic shows how to associate a Global Access Group with a learning item.

Assigning Global Access Groups to Learning Items

This example GlobalAccessGroupRelation.dat file associates a learning item in the catalog with a Global Access Group.

Example of Loading a Learning Record Activity Attempt

A learning record activity attempt records a learner’s attempt to complete an activity, such as a self-paced activity, in an offering assignment. This topic shows how to load Learning Record Activity Attempt objects using HCM Data Loader.
Creating a Learning Record Activity Attempt

In this example, the learning record and activity are referenced by user keys.

<table>
<thead>
<tr>
<th>METADATA</th>
<th>LearningRecordActivityAttempt</th>
<th>LearningRecordNumber</th>
<th>ActivityNumber</th>
<th>ActivityAttemptStatus</th>
<th>ActivityAttemptNote</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>LearningRecordActivity_attempt</td>
<td>OLC109470</td>
<td>ILT-ACT-20180213-HDL</td>
<td>ORA_ASSN_TASK_COMPLETED</td>
<td>Note for ILT</td>
</tr>
</tbody>
</table>

Onsite activity

Examples of Loading Legacy Learning Items

A legacy learning item represents a completed course or offering for inclusion in a learner’s learning history. Legacy learning items have no supporting content, and you can’t assign them to other learners. This topic provides examples showing how to load Legacy Learning Item objects using HCM Data Loader.

Creating Legacy Learning Items

This example LegacyLearningItem.dat file creates a legacy learning item to appear in a learner’s learning history. This example identifies the legacy learning item by its source key.

<p>| COMMENT: Data for Business Object LegacyLearningItem |</p>
<table>
<thead>
<tr>
<th>METADATA</th>
<th>LegacyLearningItem</th>
<th>EffectiveStartDate</th>
<th>LearningItemNumber</th>
<th>Title</th>
<th>ShortDescription</th>
<th>OwnedByPersonNumber</th>
<th>SourceType</th>
<th>SourceId</th>
<th>SourceInfo</th>
<th>SourceSystemOwner</th>
<th>SourceSystemId</th>
<th>VISION</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERGE</td>
<td>LegacyLearningItem</td>
<td>2017/07/26</td>
<td>LEGACY-201710161121-HDL</td>
<td>Title: Legacy LI 201710161121</td>
<td>DescShort: LEGACY-201710161121</td>
<td>8153756</td>
<td>FUSION_HCM_LEARNING</td>
<td>201710161121</td>
<td>HCM Learn 201710161121 Source Info</td>
<td>VISION</td>
<td>201710161121</td>
<td></td>
</tr>
</tbody>
</table>

These rules apply when you create Legacy Learning Item objects using HCM Data Loader:

- The person identified by the **OwnedbyPersonNumber** attribute must exist in the target environment.
- **LearningItemNumber** is a required, alphanumeric value that identifies this record uniquely. It must not start with the characters OLC, which are reserved for Oracle Learning Cloud catalog items.
- The **effective start date** must be on or before the start date of any effective-dated record, such as a learning record, that references the legacy learning item record. By default, the **effective end date** is the end of time. You can’t change the effective start and end dates of an existing Legacy Learning Item object.

Loading Translated Names and Descriptions for Legacy Learning Items

You can translate legacy learning item names and their descriptions. Supply translation data in the LegacyLearningItemTranslation.dat file after the legacy learning items exist in the target environment. This example LegacyLearningItemTranslation.dat file provides translations of the name and description of an existing Legacy Learning Item object. It references the object by its source key.

| COMMENT: Data for Business Object LegacyLearningItemTranslation |
Examples of Loading Offering Custom Pricing Objects

Offering Custom Pricing objects are custom pricing rules for a course offering. This topic shows how to load Offering Custom Pricing objects using HCM Data Loader.

Creating Offering Custom Pricing Objects

This example OfferingCustomPricing.dat file identifies the Offering Custom Pricing object by its user key.

```
METADATA|OfferingCustomPricing|EffectiveStartDate|PricingRuleNumber|CurrencyCode|OfferingNumber
MERGE|OfferingCustomPricing|2018/02/04|PR-201802041723-HDL|USD|OFFERING-201802041712-HDL
```

Creating Offering Custom Pricing Components

This example OfferingCustomPricing.dat file loads two Offering Custom Pricing Components, which are identified by their user keys.

```
METADATA|OfferingCustomPricingComponent|EffectiveStartDate|PricingComponentNumber|PricingComponentType|Price|IncludeInSelfServicePricing|PricingRuleId|PricingRuleNumber
MERGE|OfferingCustomPricingComponent|2018/02/04|PC-201802041724-HDL|ORA_LIST_PRICE|100|N|PR-201802041723-HDL
MERGE|OfferingCustomPricingComponent|2018/02/04|PC-201802041725-HDL|ORA_LIST_PRICE_ADJ|200|N|PR-201802041723-HDL
```

Valid **PricingComponentType** values are defined in the ORA_WLF_PRICING_TYPE lookup type. The following values are predefined:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORA_LISTPRICE</td>
<td>List price</td>
</tr>
<tr>
<td>ORA_LISTPRICE_ADJ</td>
<td>List price adjustment</td>
</tr>
</tbody>
</table>

Example of Loading a Specialization

A specialization groups multiple courses together to help learners achieve goals that a single course can’t deliver. It contains one or more specialization sections, which group related courses, or activities, in the specialization. This topic shows how to load Specialization objects using HCM Data Loader.
Creating Specializations

This example Specialization.dat file creates a Specialization object, two Specialization Section components, and two Specialization Section Activity components. The Specialization object is identified using source keys.

COMMENT Specialization object
METADATA|Specialization|EffectiveStartDate|SpecializationNumber|Title|ShortDescription|SourceType|SourceId|SourceInfo|OwnedByPersonNumber|SourceSystemOwner|SourceSystemId
MERGE|Specialization|2018/03/05|DSS-2018043009000-HDL|Title: DSS HDL Specialization 2018043009000|SD: DSS HDL Specialization 2018043009000 Short Description|FUSION_HCM_LEARNING|2018043009000|HCM Learning Source Information for 2018043009000|8153756|VISION|2018043009000

COMMENT Specialization Section components
METADATA|SpecializationSection|EffectiveStartDate|SpecializationSectionNumber|Title|SectionPosition|NumberOfActivitiesToComplete|InitialAssignmentStatusOfActivities|SpecializationNumber
MERGE|SpecializationSection|2018/03/05|Sec-2018043009000-HDL|Title: DSS HDL Section# 1|1|1|0|DSS-2018043009000-HDL
MERGE|SpecializationSection|2018/03/05|Sec-2018043009001-HDL|Title: DSS HDL Section# 2|2|2|0|DSS-2018043009000-HDL

COMMENT Specialization Section Activity components
METADATA|SpecializationSectionActivity|EffectiveStartDate|ActivityNumber|ActivityPosition|SpecializationSectionNumber|LearningItemNumber
MERGE|SpecializationSectionActivity|2018/03/05|SecAct-2018043009000-HDL|1|Sec-2018043009000-HDL|OLC900002
MERGE|SpecializationSectionActivity|2018/03/05|SecAct-2018043009001-HDL|1|Sec-2018043009001-HDL|COURSE-2018040910351-HDL

These rules apply:

- By default, the `NumberOfActivitiesToComplete` attribute of the Specialization Section component is -1, which means that all activities in the section must be completed.
- Specialization objects that you create using HCM Data Loader have default access values as defined in the application on the Manage Default Access page.

Related Topics

- Specializations
Example of Loading Accrual Plan Enrollments

You can load absence data for plan enrollment details, such as new enrollments or updates to existing enrollments. Use the Accrual Plan Enrollment business object to:

- Enroll all employees of an organization into an accrual plan in bulk, instead of individually.
- Update employee details when there is a change in their work schedule. You can terminate their enrollment for the previous plan and enroll them into new plans according to the changed work schedule.
- Delete an employee's plan enrollment if there is no accrual record for the corresponding plan.

Preparing to Load Accrual Plan Enrollment Object

Before you load accrual plan enrollment, you must:

- Verify that there are no overlapping enrollment records for the same assignment and plan
- Ensure that the start date is within the employment dates for the employer
- Ensure that both the unit of measure and number of units are specified
- Verify that the plan type is Accrual. This loader does not support other plan types

Loading an Accrual Plan Enrollment

This example PersonAccrualPlanEnrollment.dat file uses user keys to enroll an employee into an accrual plan.

Example of Loading Absence Qualification Plan Entitlement

Using HCM Data Loader, you can load absence entitlement for qualification plans using the Absence Qualification Plan Entitlement business object:

Preparing to Load Absence Qualification Plan Entitlement

Before you load accrual balance details, you must:

- Ensure that the start date and end date are provided because the start and end dates determine entitlement calculation.
- Ensure that the value is negative when loading used entitlements.
Loading an Absence Qualification Plan Entitlement

This example PersonEntitlementDetail.dat file uses source keys to load employee used entitlement details of 5 hours for a sick leave qualification plan.

```
MERGE | PersonEntitlementDetail | VISION | ABS02220160411 | 3423322 | SICK_LEAVE | 3423322WT | 2016/04/11 | 2016/12/31 | 100 | -5
```

Guidelines for Loading Absence Entry

Using HCM Data Loader, you can load data for the following components of the Absence Entry business object:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absence Entry</td>
<td>You can load absences for:</td>
</tr>
<tr>
<td></td>
<td>• A time-based schedule: A time-based work schedule has a fixed work day pattern. For example, you define an 8-hour schedule, 5 days a week. You can create a time-based work schedule that starts at 8:00 a.m. and ends at 5:00 p.m. A worker assigned to a time-based work schedule is considered to be available for a fixed number of hours each day.</td>
</tr>
<tr>
<td></td>
<td>• An elapsed schedule: In an elapsed work schedule, workers don’t have a fixed start or end time. For example, some workers may start work at 9.00 a.m., and some at 11.00 a.m. A worker assigned to an elapsed work schedule is considered to be available for a number of hours in a day. The StartDateDuration and EndDateDuration values must be provided.</td>
</tr>
<tr>
<td>Person Childbirth or Placement Absence Record</td>
<td>You can load absence records associated with the birth or placement of a child or children in the worker’s household. This record captures planned and actual absence dates, as well as dates associated with the event, such as expected date of childbirth.</td>
</tr>
<tr>
<td>Advanced Absence Entry</td>
<td>You can load advanced absence entries when there is a breakdown of absences into specific dates. This is used when an absence is taken against different assignments or if the absence is intermittent. For example, a weekly recurring absence scheduled for a weekly visit to the doctor.</td>
</tr>
</tbody>
</table>

Preparing to Load Absence Entry Objects

Before you load absence records:

- Ensure you provide the name of the employer to verify the employment is valid as of the start date of the absence. The employee’s start date determines validation of the absence entry for the legal employer.
- Ensure that the absence type is valid as of the start date of the absence and belongs to the same legislative data group as the person the absence is for.
- Ensure that the absence reason is defined, if it’s specified as a required attribute in the absence type.

Deleting Absence Entry Records

You can also delete the Absence Entry business object using HCM Data Loader. When you delete an Absence Entry record, the child records are also deleted.
Examples of Loading Absence Entry

These example AbsenceEntry.dat files show how to load absence entries using HCM Data Loader for:

- A time based absence entry
- An elapsed absence entry
- A maternity absence entry
- An advanced absence entry

Loading Absence Entry

Example 1: Time Based Absence Entry

This example uses user keys to create an absence entry for two days sick leave.

```
METADATA|PersonAbsenceEntry|Employer|PersonNumber|AbsenceType|AbsenceStatus|ApprovalStatus|StartDate|StartTime|EndDate|EndTime|Comments|AbsenceReason
MERGE|PersonAbsenceEntry|Vision Corporation|300100005197277|SICK_LEAVE|SUBMITTED|APPROVED|2018/09/24|8:30|2018/09/25|17:00|SICK LEAVE|SICK_REASON
```

Example 2: Elapsed Absence Entry

This example uses user keys to create an absence entry for two days sick leave.

```
METADATA|PersonAbsenceEntry|PersonNumber|AbsenceType|AbsenceStatus|ApprovalStatus|StartDate|StartTime|EndDate|EndTime|Comments|StartDateDuration|EndDateDuration|AbsenceReason
MERGE|PersonAbsenceEntry|40010|10026712115|BAT_ABS_TL_Sick|SUBMITTED|APPROVED|2018/01/22||2018/01/23||Sick Comments|8|8
```

Loading Person Maternity Absence Entry

This example uses source keys to create an absence entry for maternity leave.

```
METADATA|PersonAbsenceEntry|SourceSystemOwner|SourceSystemId|Employer|PersonId(SourceSystemId)|AbsenceType|AbsenceStatus|ApprovalStatus|StartDate|StartTime|EndDate|EndTime
MERGE|PersonAbsenceEntry|VISION|AbsMaternity_HDL022|Vision Corporation|HDL022|Maternity Type|SUBMITTED|APPROVED|2018/01/26|08:00|2018/02/14|17:00
METADATA|PersonMaternityAbsenceEntry|SourceSystemOwner|SourceSystemId|PerAbsenceEntryId(SourceSystemId)|PlannedStartDate|PlannedReturnDate|LeaveDuration|IntendToWork|ExpectedDateOfChildBirth|ExpectedEndDate|ActualStartDate|ActualReturnDate|ActualDuration|ActualChildBirthDate|OpenEndedFlag
MERGE|PersonMaternityAbsenceEntry|VISION|AbsMaternityDet_HDL022|AbsMaternity_HDL022|2018/01/26|2018/02/14|2018/01/25|2018/02/14|2018/01/27|
```

Loading Person Absence Entry Detail

This example uses source keys to create an advanced absence entry.
<table>
<thead>
<tr>
<th>METADATA</th>
<th>PersonAbsenceEntry</th>
<th>Employer</th>
<th>PersonNumber</th>
<th>AbsenceType</th>
<th>AbsenceStatus</th>
<th>ApprovalStatus</th>
<th>StartDate</th>
<th>StartTime</th>
<th>EndDate</th>
<th>EndTime</th>
<th>Comments</th>
<th>SourceSystemId</th>
<th>SourceSystemOwner</th>
<th>AbsenceReason</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Vision Corporation</td>
<td>955160008192206</td>
<td>#SK_QUAL_HRS</td>
<td>SUBMITTED</td>
<td>APPROVED</td>
<td>2018/11/06</td>
<td>21:00</td>
<td></td>
<td></td>
<td></td>
<td>Adv night shift</td>
<td>1SCH_SEMP_001111121</td>
<td>VISION</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>METADATA</th>
<th>PersonAbsenceEntryDetail</th>
<th>Employer</th>
<th>PersonNumber</th>
<th>AbsenceType</th>
<th>PerAbsenceEntryDtlId</th>
<th>PerAbsEntryId(SourceSystemId)</th>
<th>AbsenceDate</th>
<th>StartTime</th>
<th>EndDate</th>
<th>EndTime</th>
<th>RowSeq</th>
<th>AbsenceStartDate</th>
<th>AbsenceStartTime</th>
<th>SourceSystemId</th>
<th>SourceSystemOwner</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Vision Corporation</td>
<td>955160008192206</td>
<td>#SK_QUAL_HRS</td>
<td>1SCH_SEMP_001111121</td>
<td>2018/11/06</td>
<td>21:00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2018/11/06</td>
<td>21:00</td>
<td>HDLERDTL05011121</td>
<td>VISION</td>
</tr>
</tbody>
</table>
22 Loading Benefits Objects

Example of Loading Beneficiary Enrollments

Use the Beneficiary Enrollment business object to designate beneficiaries of multiple employees into a benefit offering. You typically designate beneficiaries for benefit plans that have payouts in the event of a death or an accident. Similar to participant enrollment, you can load beneficiary data in bulk when you are migrating from a legacy system during a merger or acquisition.

Considerations
You must consider the following points before you load beneficiary enrollment.

- You can load the Beneficiary Enrollment object using either the default benefits relationship or the unrestricted benefits relationship. However, you cannot use both relationships simultaneously.
- You can create and update records using Beneficiary Enrollment HCM Data Loader. You can’t delete records.

Preparing to Load Beneficiary Enrollment Object
Before you can load beneficiary enrollment, you must:

- Verify that the data required for enrollment such as employee name, address, salary, and contact details are available in the HR application. It might be difficult to verify errors in HR data after the upload.
- Ensure that you do not include certifications when you are migrating data from a legacy system. This might cause completed enrollments to be suspended again. Avoiding certifications also improves performance and reduces pending action items.
- Determine the number of unique employees and the total number of enrollments that you plan to load into the application. You can obtain this information from the extracted source data.
- Ensure that the total payout percentage assigned to the beneficiaries add up to 100%. Both primary and contingent beneficiaries must have 100% assigned to them.
- Create electable choices on the basis of eligibility rules. To verify whether your electable choices follow eligibility rules, use random checks based on eligibility criteria.
- Create benefits relationships for all the employees whose data you want to load.
- Ensure that the benefit relationships are properly aligned with the hire and assignment dates. For instance, the date on which the life event is processed must follow benefit relationship date, and the benefit relationship date must be the same as or follow the hire date.
- Verify that the total number of potential life events is equal to the total number of employees whose data you want to upload. In the Evaluation and Reporting work area, Processes tab, use the Assign Corrective Life Event process to create the data related to potential life events.
- Run the participation evaluation process without applying the defaults and ensure that the number of started events match with the number of employees you determined in the previous task. You must fix any errors you encounter at this stage.
- Ensure that values are not added to the CloseLifeEvent attributes in the .dat file. You must first verify the enrollment data and then use the Close Life Event process to close a life event.
Generating the .dat File

This example BeneficiaryEnrollment.dat file designates a beneficiary into a benefits program.

Example of Loading Dependent Enrollments

Use the Dependent Enrollment business object to designate dependents of multiple employees into a benefit offering. You typically perform this task when you are migrating from a legacy system during a merger or acquisition. You can also use the data loader if you want to move dependents in bulk from one plan to another.

Considerations

You must consider the following points before you load dependent enrollment.

- You can load the Dependent Enrollment object using either the default benefits relationship or the unrestricted benefits relationship. However, you cannot use both relationships simultaneously.
- You can create and update records using Dependent Enrollment HCM Data Loader. You can’t delete records.
- Dependent Enrollment processing does not depend on Effective date, but on Life Event date.

Preparing to Load Dependent Enrollment Object

Before you can load dependent enrollment, you must:

- Verify that the data required for enrollment such as employee name, address, salary, and contact details are available in the HR application. It might be difficult to verify errors in HR data after the upload.
- Ensure that you do not include certifications when you are migrating data from a legacy system. This might cause completed enrollments to be suspended again. Avoiding certifications also improves performance and reduces pending action items.
- Determine the number of unique employees and the total number of enrollments that you plan to load into the application. You can obtain this information from the extracted source data.
- Ensure that the total payout percentage assigned to the beneficiaries add up to 100%. Both primary and contingent beneficiaries must have 100% assigned to them.
- Create electable choices on the basis of eligibility rules. To verify whether your electable choices follow eligibility rules, use random checks based on eligibility criteria.
- Create benefits relationships for all the employees whose data you want to load.
- Ensure that the benefit relationships are properly aligned with the hire and assignment dates. For instance, the date on which the life event is processed must follow benefit relationship date, and the benefit relationship date must be the same as or follow the hire date.
- Verify that the total number of potential life events is equal to the total number of employees whose data you want to upload. In the Evaluation and Reporting work area, Processes tab, use the Assign Corrective Life Event process to create the data related to potential life events.
• Run the participation evaluation process without applying the defaults and ensure that the number of started events match with the number of employees you determined in the previous task. You must fix any errors you encounter at this stage.

• Ensure that values are not added to the CloseLifeEvent attributes in the .dat file. You must first verify the enrollment data and then use the Close Life Event process to close a life event.

Generate the .dat File
This example DependentEnrollment.dat file designates a dependent into a benefits program.

```
METADATA|DependentEnrollment|PersonNumber|BenefitRelationship|EffectiveDate|LifeEvent|LifeEventOccuredDate
MERGE|DependentEnrollment|10026712115|Default|2016/10/28|NewHire|2016/10/28
METADATA|DesignateDependent|Plan|Program|Option|DependentPersonNumber|LineNumber|PersonNumber
MERGE|DesignateDependent|GlobalPlan|Green Company Full Benefits|Employee plus Family|100010026713030|1|10026712115
```

Example of Loading Participant Enrollments
Use the Participant Enrollment business object to enroll participants into a benefit offering in bulk.

Considerations
You must consider the following points before you load participant enrollment.

• You can load the Participant Enrollment object using either the default benefits relationship or the unrestricted benefits relationship. However, you cannot use both relationships simultaneously.

• You can create records using Participant Enrollment HCM Data Loader. You can’t update or delete records.

• In order to update a record, de-enroll the participant first and then re-enroll the participant with the new value. Refer to the example ParticipantEnrollment.dat files provided in this topic to de-enroll and re-enroll a participant.

Preparing to Load Participant Enrollment Object
Before you can load participant enrollment, you must:

• Verify that the data required for enrollment such as employee name, address, salary, and contact details are available in the HR application. It might be difficult to verify errors in HR data after the upload.

• Ensure that you do not include certifications when you are migrating data from a legacy system. This might cause completed enrollments to be suspended again. Avoiding certifications also improves performance and reduces pending action items.

• Determine the number of unique employees and the total number of enrollments that you plan to load into the application. You can obtain this information from the extracted source data.

• Ensure that the total payout percentage assigned to the beneficiaries add up to 100%. Both primary and contingent beneficiaries must have 100% assigned to them.

• Create electable choices on the basis of eligibility rules. To verify whether your electable choices follow eligibility rules, use random checks based on eligibility criteria.

• Create benefits relationships for all the employees whose data you want to load.

• Ensure that the benefit relationships are properly aligned with the hire and assignment dates. For instance, the date on which the life event is processed must follow benefit relationship date, and the benefit relationship date must be the same as or follow the hire date.
• Verify that the total number of potential life events is equal to the total number of employees whose data you want to upload. In the Evaluation and Reporting work area, Processes tab, use the Assign Corrective Life Event process to create the data related to potential life events.

• Run the participation evaluation process without applying the defaults and ensure that the number of started events match with the number of employees you determined in the previous task. You must fix any errors you encounter at this stage.

• Ensure that values are not added to the CloseLifeEvent attributes in the .dat file. You must first verify the enrollment data and then use the Close Life Event process to close a life event.

Generating the .dat File
This example ParticipantEnrollment.dat file enrolls a participant into a benefits program.

```
METADATA|ParticipantEnrollment|PersonNumber|ParticipantLastName|ParticipantFirstName|BenefitRelationship|LifeEvent|LifeEventOccuredDate|EffectiveDate
MERGE|ParticipantEnrollment|10026712115|Duprey|Luis|Default|NewHire|2016/10/28|2016/11/05
METADATA|CompensationObject|Program|OriginalEnrollmentDate|PersonNumber|LineNumber
MERGE|CompensationObject|Green Company Full Benefits|2014/10/28|10026712115|1
```

**Note:** It is not necessary to supply the ParticipantLastName and ParticipantFirstName attributes. The application does not perform any validation to cross check the name against the person number. These attributes are for provided only for your information.

De-enrolling from an Existing Option in the Plan and Enrolling in a Waive Option
You can de-enroll employees from a plan and enroll them into the waive option of same plan type. For instance, consider an employee who is enrolled in the Healthy HMO plan with the Employee Plus Family option. You can de-enroll the employee from the Employee Plus Family option and enroll the employee in the Waive Coverage option using the following example ParticipantEnrollment.dat file.

```
METADATA|ParticipantEnrollment|PersonNumber|ParticipantLastName|ParticipantFirstName|BenefitRelationship|LifeEvent|LifeEventOccuredDate|EffectiveDate
MERGE|ParticipantEnrollment|10026712115|Duprey|Luis|Default|Gain Dependent|2016/10/28|2016/10/28
METADATA|CompensationObject|Plan|Program|Option|OriginalEnrollmentDate|PersonNumber|DenrollPlan|DenrollOption|LineNumber
MERGE|CompensationObject|Healthy HMO|Healthy Benefits|Waive Coverage|2016/10/28|10026712115|Healthy HMO|Employee plus Family|1
```

Example of Loading Person Benefit Group
You create a benefit group to determine a specific set of people who are eligible for a benefit object. Use the Person Benefit Group business object to load multiple benefits groups simultaneously using HCM Data Loader.

Preparing to Load Person Benefit Group Object
Verify that the data required for loading a business group, such as the person name, is available in the HR application. It might be difficult to verify errors in HR data after the upload.
Generating the .dat File

This example PersonBenefitGroup.dat file loads multiple benefit groups.

```
METADATA|PersonBenefitGroup|SourceSystemOwner|SourceSystemId|BenefitGroupName|EffectiveStartDate|PersonId
MERGE|PersonBenefitGroup|VISION|GROUP955160008173169|VISION_BENEFIT_GROUP|2019/01/01|955160008173160
MERGE|PersonBenefitGroup|VISION|GROUP100000012556816|VISION_BENEFIT_GROUP|2019/01/01|100000012556816
```

Example of Loading Person Benefit Balance

Use the Person Benefit Balance business object to load multiple benefits balances using HCM Data Loader. You typically use benefits balances to store benefit amounts from an external system, or as a workaround for a calculation to bridge disparate systems.

Preparing to Load Person Benefit Balance Object

Verify that the data required for uploading benefit balances such as benefit balance, legal entity, and assignment details are available in the HR application. It might be difficult to verify errors in HR data after the upload.

Generate the .DAT File

This example PersonBenefitBalance.dat file loads a person benefit balance.

```
METADATA|PersonBenefitBalance|SourceSystemOwner|SourceSystemId|PersonId|BenefitBalanceName|BenefitRelationName|EffectiveStartDate|UOM|Val
MERGE|PersonBenefitBalance|VISION|BALANCE955160008173169|955160008173169|SICK_LEAVE_BALANCE|DFLT|2009/01/01|USD|2200
```

Example of Loading Person Habits

Use the Person Habits business object to load details of habits and disability statuses of employees in bulk using HCM Data Loader.

Preparing to Load Person Habits

Before you can load person habits, you must:

- Verify that the objects referenced by Person Habits exists in the target environment. If they do not exist, the load will fail.
- Create electable choices on the basis of eligibility rules. Lack of eligibility causes a data roll back for corresponding employees. To verify whether your electable choices follow eligibility rules, use random checks based on eligibility criteria.

Generating the .dat File

This example PersonHabits.dat file loads the person habits for an employee who is a full time student, smokes and is registered as disabled.

```
METADATA|PersonHabits|SourceSystemOwner|SourceSystemId|PersonNumber|EffectiveStartDate|LegalEmployer|DisabilityStatus|RegisteredDisabledFlag|StudentStatus|TobaccoTypeUsage
```
| MERGE | PersonHabits | VISION | HABIT955160008173169 | 955160008173169 | 2015/01/01 | Vision_Corporation | A | Y | FULL_TIME | Y |
23 Importing and Loading Data Using HCM Data Loader

How Data Is Imported and Loaded

HCM Data Loader imports data initially to stage tables, where some validation occurs. HCM Data Loader then loads valid logical objects from the stage tables to the application tables. You can perform the import and load steps either separately or as one step. This topic summarizes the processing that occurs in each step. It also describes how you specify import and load options.

Importing Records to the Stage Tables

In the import stage, HCM Data Loader:

- Extracts the business-object data from your .zip file on the Oracle WebCenter Content server and decrypts it, if the file is encrypted.
- Validates the business-object file name and METADATA definitions and raises any errors.
- Imports records from the data files into the HCM Data Loader stage tables.
- Validates the data against attribute data types and raises any errors.
- Groups the data by local key values to form logical records of related date-effective file lines. For example, all date-effective file lines supplied for a Job object are grouped into a logical occurrence of a job, such as the Accountant job.
- Forms logical objects by resolving references to parent components. For example, the logical record for a valid grade is associated with its parent Job object.
- Records the status of all imported objects. View the status when you click the Total Objects count from the Import and Load status page.

Loading Objects to the Application Tables

HCM Data Loader doesn’t load valid logical objects directly into the application tables. Instead, it passes valid object data to business-object-specific services. For example, to load Worker objects, HCM Data Loader passes them to the worker service. These services perform the validation that’s specific to the business object, raise any errors, and load valid objects to the application tables.

Setting the File Action Parameter

When you import a file on the Import and Load data page, you can set the File Action parameter on the Schedule Request page. The value that you specify here overrides the enterprise default setting. This table describes how the File Action parameter controls import and load.

<table>
<thead>
<tr>
<th>File Action Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import and load</td>
<td>HCM Data Loader imports the contents of the .zip file to the stage tables and calls object services to load valid logical objects to the application tables.</td>
</tr>
</tbody>
</table>
You may want to select **Import only** when you start to load data so you can fix import errors before attempting to load objects. Once your data loading is routine, use **Import and load** to run them both with one job submission.

**Tip:** You can validate your data file before you load it. Use the Data File Validator tool for HCM Data Loader. For more information, see Data File Validator Tool for HCM Data Loader (2022617.1) on My Oracle Support at https://support.oracle.com.

### Automating Data Loading

When you automate data loading using the HCM Data Loader web service, you can set **File Action** to **Import only** or **Import and load**. If you select **Import only**, then you load the imported objects manually on the Import and Load Data page.

### Reviewing Object History

For some object components, such as Assignment, you can review the object history to see who last updated the object. When the object was last updated using HCM Data Loader, the **Last Updated By** value is FUSION_APPS_HCM_ESS_LOADER_APPID. You don’t see the user name of the user who uploaded the changes.

### Import and Load Data

The file of data that you’re preparing to import and load using HCM Data Loader may exist locally on your desktop. Or, you may already have loaded it to the Oracle WebCenter Content server. This topic provides steps for importing and loading your data interactively on the Import and Load Data page. It also describes how to load data to the application tables if you imported it separately to the stage tables.

### Import and Load a Local File

1. On the Tasks panel tab of the Data Exchange work area, select **Import and Load Data** to open the Import and Load Data page.
2. Click **Import File**.
3. In the Import File dialog box, click **Browse** to search for and select your file. Alternatively, drag the file from your local file browser to the **Browse** button.
4. Click **Submit** to upload your file to the Oracle WebCenter Content server. The file is loaded automatically to the HCM Data Loader import account hcm/dataloader/import and allocated a unique content ID.
5. In the Parameters section of the Import and Load Data page, review the parameter values and update them as required. Most of these values are set for the enterprise on the Configure HCM Data Loader page and can be left unchanged. The parameters are shown in this table.
### Parameter | Description
--- | ---
**File Name** | The name of the file on the WebCenter Content server.

**Content ID** | The WebCenter Content ID for the file on the WebCenter Content server.

**File Action** | The *Import only* option imports the business object data into the stage tables but doesn't load it to the application tables. You can start the load process manually from the user interface.

The *Import and load* option automatically loads to the application tables any business object that loads successfully to the stage tables.

**File Encryption** | The encryption type for the file.

**Maximum Percentage of Import Errors** | The percentage of file lines in error that can occur in a business object before the import process stops for the object.

**Maximum Percentage of Load Errors** | The percentage of business-object instances in error that can occur for a business object before the load process stops.

**Delete Source File** | Indicates whether the source file is to be purged from the WebCenter Content server after processing.

**Maximum Concurrent Threads for Import** | The maximum number of concurrent process threads to use for importing your data set.

**Maximum Concurrent Threads for Load** | The maximum number of concurrent process threads to use for loading your data set.

**Load Group Size** | The number of objects to process at a time on each thread. The record counts are updated only when processing completes for each group.

*Note:* This parameter appears on the Schedule Request page only if the *Enable Load Group Size Calculation* configuration parameter is set to *No*. Otherwise, the load-group size is calculated automatically.

6. Click **Submit**. A confirmation message showing the process ID appears.
7. Note the process ID and click **OK** to close the Confirmation dialog box.

---

**Import a File from the Oracle WebCenter Content Server**

You can load a .zip file to the Oracle WebCenter Content server if you prefer. Select **Navigator > Tools > File Import and Export** and upload the file to the hcm/dataloader/import account. To import and load a file that’s already on the WebCenter Content server:

1. On the Tasks panel tab in the Data Exchange work area, select **Import and Load Data** to open the Import and Load Data page.
2. On the Import and Load Data page, click **Import File**.
In the Import File dialog box, select **Oracle WebCenter Content Files**. Any files in the hcm/dataloader/import account that haven’t been processed are listed. You can filter the list to reduce it. For example, you can enter the file creation date.

3. Select your file in the Import File dialog box and click **Submit**.
4. In the Parameters section of the Import and Load Data page, review the parameters, edit them as necessary, and click **Submit**.

**Start or Resubmit a Load**

You can start the load process directly for a business object. You may want to do this because you submitted your data set in **Import only** mode. Or, you may want to resubmit a failed load process.

1. In the Business Objects table on the Import and Load Data page, select the business object and click **Load**.

   ✷ **Note:** The **Load** button is enabled only for business objects with a **Loaded** status of **Ready**, **Error**, or **Stopped**.

2. On the Schedule Request page, review the parameter values, edit them if necessary, and click **Submit**.

**Upload Data Using HCM Data Loader**

In this example, you learn how to import and load objects manually to Oracle HCM Cloud. Import and load are performed separately in this example.

This table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which business object am I loading?</td>
<td>Job. The child components of the Job object aren’t being loaded. These components are:</td>
</tr>
<tr>
<td></td>
<td>• Job Evaluation</td>
</tr>
<tr>
<td></td>
<td>• Job Extra Information</td>
</tr>
<tr>
<td></td>
<td>• Job Legislative Extra Information</td>
</tr>
<tr>
<td></td>
<td>• Job Valid Grade</td>
</tr>
<tr>
<td>What’s the template name?</td>
<td>Job.dat</td>
</tr>
<tr>
<td>What’s the name of the file that I am uploading?</td>
<td>Sales_Jobs.zip</td>
</tr>
<tr>
<td>What's the value of the File Action parameter?</td>
<td><strong>Import only</strong>. You have decided to manage the load step separately.</td>
</tr>
</tbody>
</table>
Summary of Tasks

Upload the jobs data by:

1. Generating the jobs template
2. Editing the template file
3. Importing jobs to the stage tables
4. Loading jobs to the application tables

Generate the Job Template

1. On the Tasks panel tab of the Data Exchange work area, select View Business Objects.
2. On the View Business Objects page, enter Job in the Business Object query-by-example field and press Enter. Select the row for the Job object in the Search Results section.
3. Click Refresh Object on the table toolbar.
4. In the Schedule Request dialog box, click Submit.
5. Click OK to close the confirmation message.
6. Click Refresh on the table toolbar to view the Download File icon in the File column for the job row.

You may have to refresh more than once before the icon appears.

Tip: If the icon already exists, then you can download the existing template. The Last Refreshed column shows when the template was generated.

7. Click the Download File icon in the File column.
8. Save the Job.dat template locally.

Edit the Template File

1. Open the Job.dat template.
2. Edit the template file as follows:
   - Remove METADATA lines for these unwanted child components: Job Evaluation, Job Extra Information, Job Legislative Extra Information, and Job Valid Grade.
   - Edit the job METADATA line to remove unwanted attributes.
   - Make any required changes to the attribute order.
3. Enter jobs data in the configured template. For example:
   METADATA|Job|SourceSystemOwner|SourceSystemId|EffectiveStartDate|SetCode|JobCode|Name|ActiveStatus|FullPartTime|RegularTemporary|JobFamilyId(SourceSystemId)|JobFunctionCode|ManagerLevel
   MERGE|Job|VISION|SALES_CONS|2000/01/01|COMMON|SALES_CONS|Sales Consultant|A|FULL_TIME|R|CONSULT||9
   MERGE|Job|VISION|SALES_MGR|2000/01/01|COMMON|SALES_MGR|Sales Manager|A|FULL_TIME|R|MANAGEMENT|MANAG|6
   MERGE|Job|VISION|SALES_DIR|2000/01/01|COMMON|SALES_DIR|Sales Director|A|FULL_TIME|R|MANAGEMENT|MANAG|1
   MERGE|Job|VISION|MRKT_CONS|2000/01/01|COMMON|MRKT_CONS|Marketing Consultant|A|FULL_TIME|R|CONSULT||9
   MERGE|Job|VISION|MRKT_MGR|2000/01/01|COMMON|MRKT_MGR|Marketing Manager|A|FULL_TIME|R|MANAGEMENT|MANAG|6
   MERGE|Job|VISION|MRKT_DIR|2000/01/01|COMMON|MRKT_DIR|Marketing Director|A|FULL_TIME|R|MANAGEMENT|MANAG|1
4. Save and close the template file.
5. Compress the template file, name it `Sales_Jobs.zip`, and save it locally.

### Import Data to the Stage Tables

1. On the Tasks panel tab in the Data Exchange work area, select **Import and Load Data**.
2. On the Import and Load Data page, select **Import File**.
3. In the Import File dialog box, click **Browse**.
5. Click **Submit**.
6. On the Schedule Request page, select **Import only** in the **File Action** field.
7. Click **Submit**.
8. Click **OK** to close the confirmation message.
9. On the Import and Load Data page, click **Refresh** in the **Data Sets** table.
10. Review the status and details of the `Sales_Jobs.zip` data set that you uploaded. When the Total Objects count is greater than zero you can click the new count to review the objects found in your file and their status. Refresh the page if the data set is still in progress. Click Object Details to review the object hierarchy and the values of attributes supplied for the logical object.

Return to the Data Sets table on Load and Import Data page. When the **Import Status** column shows **Success** and the **Import Success (%)** column displays **100**, you’re ready to load the data to the application tables.

### Load Data to the Application Tables

1. In the Data Sets table on the Import and Load page, select the `Sales_Jobs.zip` data set.
2. In the Business Objects table on the Import and Load Data page, select the Job object row and click **Load**.
3. In the Schedule Request dialog box, click **Submit**.
4. Click **OK** to close the confirmation message.
5. Click **Refresh** on the Business Objects table toolbar to check the status of the load.

When the **Load Status** column shows **Success** and the **Load Success (%)** column displays **100**, the job data has loaded successfully to the application tables.

### How You Review and Correct Errors

Errors that occur during the import and load stages of data loading are reported automatically on the Import and Load Data page. This topic describes how to review and correct reported errors.

#### Reviewing Import Errors

The **Failed Lines** column in both the Data Sets and Business Objects tables shows the number of records that failed to import. When the number in the **Failed Lines** column is greater than zero, clicking it opens the Failed Lines page. When you open the Failed Lines page from the Data Sets table, it lists:

- The METADATA line for each component
- All messages and the file lines in error for the selected data set
You can:

- Filter the list to show the METADATA lines and the file lines in error for the components of a single business object.
- Use Query by Example fields to show specific file lines or messages.

When you open the Failed Lines page from the Business Objects table, you see details for the selected object by default. You can filter the list to show failed lines for different objects from the same data set.

Examples of physical row errors are:

- The instruction or discriminator isn’t recognized.
- The number of values in a data line doesn’t match the number in the METADATA line.
- You didn’t supply a unique reference for the component.
- The parent of a child component wasn’t found, either in the file or in Oracle HCM Cloud.
- A required attribute is missing.

To correct import errors, you correct the source data before importing and loading it again. You need to add the corrected data file to a new .zip file. You can't process the existing file again.

### Reviewing Load Errors

The Failed Objects column in both the Data Sets and Business Objects tables shows the number of objects that failed to load. When the number in the Failed Objects column is greater than zero, clicking it opens the Failed Objects page. When you open the Failed Objects page from the Data Sets table, it includes for each object in error:

- Its user keys and any source key
- The error message
- A link to the object details, which opens the Object Hierarchy page

You can:

- Filter the list to show error details for a specific object in the data set.
- Use Query by Example fields to show specific keys or messages.

When you open the Failed Objects page from the Business Objects table, you see details for the selected object by default. You can filter the list to show failed lines for different objects from the same data set.

### Reviewing Messages

To review the messages for an import or load failure, click the icon in the Messages column of the Data Sets or Business Objects table. When you open the Messages page for a data set, you see for each message:

- The process where the message occurred, which can be Import or Load
- The message location
- The message itself, with the number of occurrences of the message

When you click the message link, you open the Message Details page. It provides details of the message with the information from the Failed Lines or Failed Objects page, as appropriate.

**Tip:** Messages appear in the session language if the HCM Data Loader configuration parameter Enable UI Message Translation is set to Yes in your environment.
Correcting Objects in Error

On the Object Hierarchy page, which you open from the Failed Object, Message Detail or Object Status page, you can expand object folders to see error details. When you click the name of a component entry, the Attribute Details dialog box opens. It shows the original and current values of each attribute. For components in error, you can edit an attribute to correct it before loading the object again on the Import and Load Details page. You can see how many objects you corrected in the Objects Corrected column of the tables on the Import and Load Data page. When you click the number in the Objects Corrected column, the Failed Objects page opens for the corrected object.

If you prefer to correct object errors in the source data, then you need to import and load the corrected data again. You must add the corrected data file to a new .zip file. You can’t process the existing file again.

Stop HCM Data Loader Processing

You can stop the processing of either a data set or an individual business object. This topic explains how to stop processing.

Stop the Processing of a Data Set

Follow these steps:

1. In the Search Results section of the Import and Load Data page, select the data set that you want to stop.
2. Click Refresh to update the status and confirm that the data set is still processing.
3. Click Stop.

Tip: The Stop button is active only if either the import or the load is still in progress.

4. In the Warning dialog box, click Yes to confirm the stop request.

The status of the import or load process that was stopped is updated to Stopped. A large data set may take a few minutes to stop processing completely. For efficiency in import and load processing, HCM Data Loader checks at intervals for stop requests.

Stop the Processing of a Business Object

Follow these steps:

1. In the Details section of the Import and Load Data page, select the business object that you want to stop processing.
2. Click Refresh to update the status and confirm that the business object is still processing.
3. Click Stop.
4. In the Warning dialog box, click Yes to confirm the stop request.

The status of the import or load process that was stopped is updated to Stopped.

Restart Stopped Processes

You can’t restart a stopped import process for a business object. However, you can restart a stopped load process for a business object. Select the stopped business object in the Details section of the Import and Load Data page and click Load.

The status of both the business object and the associated data-set load processes changes to In Progress.
Overview of Automating Data Loading

You can automate HCM Data Loader processes to load data to Oracle HCM Cloud and extract both status and error information. This topic provides an introduction to the automation processes.

The processes that you can automate are:

- Delivery of .zip files to the Oracle WebCenter Content server
- Initiation of HCM Data Loader to process files on the WebCenter Content server
- Initiation of the HCM Data Loader Data Set Summary extract and retrieval of its output

Complete instructions are provided in Automating HCM Data Loader (1664133.1) on My Oracle Support at https://support.oracle.com.

Delivering Files to Oracle WebCenter Content Automatically

To automate the delivery of .zip files to the WebCenter Content server, you can use:

- The WebCenter Content Document Transfer Utility, a feature-rich Java library that provides content import and export capabilities
- WebCenter Content Generic SOAP Web Service, which exposes a SOAP web service interface (GenericSoapPort) that supports various operations, including file upload

Initiating HCM Data Loader Automatically

You can initiate HCM Data Loader automatically to process multiple files on the Oracle WebCenter Content server using the HCM Data Loader Integration Service. This web service provides the two methods shown in this table.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>importAndLoadData</td>
<td>Initiates the processing of a data file</td>
</tr>
<tr>
<td>getDataSetStatus</td>
<td>Returns the current status of a data file</td>
</tr>
</tbody>
</table>

Several ways exist of running the web service. For example, you can use generated proxy classes.

Initiating the HCM Data Loader Data Set Summary Extract Automatically

HCM Extracts is a tool for reporting and outbound integrations. When you use HCM Extracts as part of an integrated process, you can initiate it automatically from an automated flow. The Payroll Flow Engine is the generic processing engine that you use for this purpose.
When you save a copy of the predefined HCM Data Loader Data Set Summary extract, you also create a payroll flow of type Extract. You can initiate this payroll flow using the FlowActionsService, which you use to automate and monitor the extract. You use the payrollProcessingActionService.fetchExtractOutput method to download the extract output. When the extract is associated with a BI template, the payroll flow ensures that the BI report is generated and written to the BI server.

Related Topics
- Automating HCM Data Loader (1664133.1)
- HCM Data Loader Extracts

FAQs for Importing and Loading Data Using HCM Data Loader

What's an HCM Data Loader data set?
A set of data supplied together to be loaded into Oracle HCM Cloud. The source of a data set is either a compressed data file containing delimited business-object-specific data files or an HCM Spreadsheet Data Loader spreadsheet.

What happens in the import stage?
The application imports individual file lines from your compressed data file to the HCM Data Loader stage tables. Related file lines are grouped to form logical business objects, and some preliminary validation occurs. Errors reported at this stage are specific to the object itself. The object’s dependencies on other objects aren’t assessed.

What happens in the load stage?
The application validates foreign-object references and applies business-object-specific validation rules. If the object passes all validation, then it’s loaded to the Oracle HCM Cloud application tables. If any component of the object fails validation, then the whole logical object fails to load. You can monitor object status on the Import and Load Data page. Errors reported at this stage may relate to either the object itself or its dependencies on other objects.

How can I review the status of the objects in my file?
When you import data into the stage tables, HCM Data Loader records the status of all imported objects. On the Import and Load status page, the Total Objects count increases as the import process progresses. Click the count to open the Object Status page and review your objects. If the process hasn’t finished, refresh the Object Status page until all imported objects appear.
How can I view the data in error for a data set or business object?

If your data file import encounters errors, then the **Failed Lines** and **Failed Objects** columns on the Import and Load Data page will become active links. Click the number to open the Failed Lines or Failed Objects page, as appropriate, to see error details. From the Failed Objects page, you can link to the Object Hierarchy page and correct individual attribute values.

Or, you can click the **Messages** icon on the Import and Load Data page to open the Messages page. From the Messages page, you can navigate to error details. For failed objects, you can open the Object Hierarchy page.

How do I fix HCM Data Loader errors?

You need to fix import errors in the source data and upload the corrected data again in a new .zip file. You can't process the existing .zip file again.

You can also fix load errors in the source data and upload the corrected data again. Or, you can correct attribute values in the Attribute Details dialog box on the Object Hierarchy page for a failed object. In this case, you can resubmit the load process for corrected objects on the Import and Load Data page.
24 Data Loading Using Spreadsheets

Spreadsheet Data Loaders

In this topic, you learn about using the spreadsheet data loader provided with Oracle HCM Cloud. To use this tool, you must install the Oracle ADF Desktop Integration Add-in for Excel.

HCM Spreadsheet Data Loader

You can use HCM Spreadsheet Data Loader to load most objects that HCM Data Loader supports. The exceptions are components, such as Document Record Attachment and Person Image, that load content like PDF files and images into the Cloud. Here’s how the spreadsheet gets processed:

- Rows in the spreadsheet are initially saved to the HCM Spreadsheet Data Loader stage tables
- Stage table data is reformatted to the hierarchical shape supported by HCM Data Loader
- Reformatted stage table data gets saved to the HCM Data Loader staging table

You can find HCM Spreadsheet Data Loader in the Data Exchange work area.

Single-Object Spreadsheet Loaders

Oracle Fusion HCM provides many spreadsheet loaders for specific object types.

In most cases, you:

- Create the spreadsheet on the page where you manage the object.
- Load the objects directly to the application tables.
- Correct any errors in the spreadsheet.

In many cases, you can both create and update the object in a spreadsheet.

Single-object spreadsheets are most useful for users such as human resource specialists and benefits specialists, who are creating objects of a specific type in bulk. You’re less likely to use these types of spreadsheet loaders for integrations.

This table shows some key objects that you can create in bulk using this type of spreadsheet. It also identifies the page or tab where you generate the spreadsheet.

<table>
<thead>
<tr>
<th>Business Object</th>
<th>Page or Tab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benefit Enrollment</td>
<td>Evaluation and Reporting work area, Enrollment Uploads tab</td>
</tr>
<tr>
<td>Benefit Plan</td>
<td>Manage Benefit Plans page</td>
</tr>
<tr>
<td>Benefit Program</td>
<td>Manage Benefit Programs page</td>
</tr>
</tbody>
</table>

Note: Some of these are HCM Spreadsheet Data Loader spreadsheets embedded in the relevant page or tab.
For some objects, such as Benefit Rate and Assignment, you can perform a mass update in a spreadsheet. To update these objects, you download existing objects to a generated spreadsheet, edit the objects, and upload the changes.

### Set Up Desktop Integration for Excel

You can create or edit records that you can upload to the application using integrated Excel workbooks. To use these workbooks, you must install an Excel add-in.

> **Note:** Every release of Oracle Fusion Applications is likely to have a new version of the Oracle ADF Desktop Integration add-in. When prompted, install the latest version of the add-in. For more information, see Add-In Upgrade Policy for ADF Desktop Integration (2242879.1) on My Oracle Support at https://support.oracle.com.
Before You Start

- Make sure you have an Excel and Windows version that’s listed in Supported Platforms for ADF Desktop Integration (2242428.1) on My Oracle Support at https://support.oracle.com.
- If you’re reinstalling the Excel add-in and currently have a version older than 11.1.1.7.3 (4.0.0), then uninstall the existing Oracle ADF Desktop Integration Add-In for Excel the same way you uninstall any program on your computer.

**Tip:** You can find the version in the control panel where you uninstall programs.

- Optionally install the following from the Microsoft website.
  - Microsoft .NET Framework 4.5.2
  - Microsoft Visual Studio 2010 Tools for Office Runtime (VSTO Runtime)

The add-in installer does check if you have these already, and would download and install them if needed. But, you can manually install them first, especially if you run into issues installing them as part of installing the Excel add-in.

Install the Desktop Add-In

To install the Oracle ADF 11g Desktop Integration Add-In for Excel:

1. Make sure you are signed in to your computer with your account. For example, you can’t have someone else sign in as an administrator and make the installation available for everyone using your computer.
2. In the application, look for the client installer in **Navigator > Tools**.
3. Run the installer (**adfdi-excel-addin-installer.exe**) as you would any program that you install on your computer.

**Related Topics**

- Guidelines for Using Desktop Integrated Excel Workbooks

Troubleshoot Desktop Integration for Excel

The application is integrated with Microsoft Excel so that, where available, you can work with records in a desktop integrated workbook. You might run into issues with the integration, for example, if you can’t open the workbook that you downloaded or the workbook doesn’t look right. You can use the Client Health Check Tool. For more information see Information Center: Troubleshooting Oracle ADF Desktop Integration (2012600.2) on My Oracle Support at https://support.oracle.com.

Use the Client Health Check Tool

Use the health check tool to find out what integration issues you might have and how to resolve them. Ask your help desk if you are unable to find or use the tool.

1. Download the latest version of the health check tool from How to use ADF Desktop Integration Client Health Check Tool (2010222.1) on My Oracle Support at https://support.oracle.com.
2. Run **clientHealthCheck.exe** as you would other programs on your computer, and review the result for each checked item.
3. Select any item that has a problem, and read the help text.
4. Fix some of the problems by clicking the **Fix Problems** button. Otherwise, follow the instructions in the help text.
5. If you need more assistance, click the **Save Report As** button to prepare information for your help desk.
6. Review the report and remove any sensitive information.
7. Contact your help desk and provide your report.
25 Introduction to HCM Spreadsheet Data Loader

Guidelines for Using HCM Spreadsheet Data Loader

You can load data to the HCM Data Loader stage tables from either a data (.dat) file or a spreadsheet. Your approach depends largely on how often you load data and the complexity of the data that you’re loading. This topic describes some key features of data loading from spreadsheets. This information helps you to identify the best approach for your business case.

Spreadsheet Templates

You must generate the spreadsheet from a spreadsheet template. Some spreadsheet templates are predefined. To create templates, you can either copy a predefined template and edit it or create a template from scratch.

When to Use Spreadsheets

Spreadsheets offer many advantages. For example:

- Spreadsheets are easy to use.
- You can create spreadsheet templates and generate spreadsheets tailored to your business case. For example, you can omit attributes, change attribute order, change attribute labels, and add help text. You can also define different templates for creating and maintaining the same business objects.
- You can enable business users to load data for selected spreadsheet templates.
- You can save data periodically from a spreadsheet to the stage tables. Therefore, you can load large volumes of records without being prevented by limits on spreadsheet size.
- Errors are reported in the spreadsheet, and you can correct them there.
- You can mix your use of spreadsheets with use of .dat files. For example, you can load data from a spreadsheet but maintain it using .dat files.
- You can use spreadsheets to delete data.
- In specific cases, spreadsheets are the simpler option. For example, to load an element entry for an element with 5 lines from a .dat file, you supply:
  - 1 record for the element entry
  - 1 record for each name and value

In a spreadsheet, you supply just 1 spreadsheet line where the values are individual columns on the same line.

Spreadsheet Restrictions

When loading data from a spreadsheet:

- You can’t upload data to attributes whose data type is binary large object or character large object, like images and file attachments.
HCM Spreadsheet Data Loader doesn't recognize source keys. Therefore, objects that you load from a spreadsheet have default source keys. To maintain such objects using .dat files, you supply user keys.

You can't create objects or object components, such as Profile Item, that don't supply user keys. However, you can update such objects if you define a list of values on the local surrogate ID attribute to identify the record to update.

If an object has more than one user key, then one is selected automatically when you include a user-key attribute in the spreadsheet template. You can't select a different user key from the one that's selected automatically. The default user key may include attributes that you don't want to include in your spreadsheet template. If those attributes appear in all other user keys, then you must include them and specify a valid value. Otherwise, you can set them to #NULL.

**Tip:** View the object on the View Business Objects page. When an object has multiple user keys, a link replaces the list of attributes in the **User Keys** column. The link opens a dialog box. The first user key in the dialog box is the one that's used automatically in spreadsheet templates.

You can't update the values of the user-key attributes if they identify the record to update. Therefore, you're recommended to define separate spreadsheets for creating and updating objects. In the spreadsheet for updating an object, you can identify the record to update in one of these ways:

- Include the local surrogate ID attribute, which has a predefined LOV.
- Include the primary user key attribute, but configure an LOV for the attribute value. With this approach, the user sees a recognizable value in the spreadsheet.

**Note:** Whether you load data from a spreadsheet or a .dat file, no approval processes exist for loaded objects.

Setup Requirements

All users must set up desktop integration for Microsoft Excel to use HCM Spreadsheet Data Loader.

Related Topics

- HCM Spreadsheet Data Loader Templates
- Set Up Desktop Integration for Excel
- Configure Lists of Values for Attribute Validation

How Data Is Uploaded Using HCM Spreadsheet Data Loader

When you click **Upload** in a spreadsheet, your data is uploaded to Oracle HCM Cloud. Behind the scenes, upload is a two-stage process. In the first stage, HCM Spreadsheet Data Loader imports the data to the HCM Data Loader **stage tables**. In the second stage, HCM Data Loader loads valid business objects to the application tables.
This figure summarizes the process of uploading data using HCM Spreadsheet Data Loader.

How Spreadsheet Data Is Loaded

The details of this process are:

1. You use the Run Spreadsheet Data Loader task in the Data Exchange work area to generate a spreadsheet for a business object from a spreadsheet template directly. You can also access spreadsheet templates directly from the business objects pages, such as Job, Location, Position.

2. You click Create Data Set in the spreadsheet to create a data set for the data that you want to load.

3. You enter data in the spreadsheet and click Upload. This action imports the spreadsheet data set to the HCM Data Loader stage tables.

4. Some validation occurs in the stage tables. Any import errors are reported in the Progress and Message columns of the Spreadsheet Line Status section of the spreadsheet when you click Refresh.

   Errors at this stage are rare, as the spreadsheet format enforces business-object structure and rules.

5. As each object loads successfully to the stage tables, HCM Data Loader calls the relevant object service to load the object to the application tables.

6. The object service validates the object. Any errors are reported in the Progress and Message columns of the Spreadsheet Line Status section of the spreadsheet when you click Refresh.
7. You correct any import or load errors in the spreadsheet and click **Upload** again. This action imports the corrected data to the stage tables again.

**Tip:** You can review the status of an HCM Spreadsheet Data Loader data set on the HCM Data Loader Import and Load Data page. Search for data sets by file type to list those loaded from spreadsheets.

The Instruction worksheet of the spreadsheet may contain instructions on how to use the spreadsheet, while the Messages worksheet holds a summary of all import and load messages generated for this data set.

**Related Topics**
- Upload Data Using HCM Spreadsheet Data Loader

### How You Test the HCM Spreadsheet Data Loader Process Flow and Connections

Before you attempt to load your own data, you may want to test HCM Spreadsheet Data Loader end-to-end processing without creating unwanted data. This topic describes how to test HCM Spreadsheet Data Loader end-to-end processing.

**Testing End-to-End Processing**

To test end-to-end processing, you use:

- A process, **Test HCM Data Loader Process Flow and Connections**
- An object, Example Object

When you run the **Test HCM Data Loader Process Flow and Connections process**, you set the **Action** parameter to **Delete test data**. The process:

1. Generates a .csv file of data for the example object
2. Initiates HCM Spreadsheet Data Loader to import and load the file
3. Creates a log file, which you can access on the Scheduled Processes page

Approximately 200 hundred records of example data are imported and loaded to the object’s own tables. No data is saved to the application tables.

**The Example Object**

The example object:

- Has child and grandchild components
- Supports create and update actions
- Has attributes of most data types

A predefined spreadsheet template is provided for the example object. The generated .csv file is based on this template.

**Deleting the Test Data**

You’re recommended to run the **Test HCM Data Loader Process Flow and Connections** process periodically to delete the test data and associated data sets. When you run the process, you set the **Action** parameter to **Delete test data**. The
process deletes all example data, regardless of whether the data was created by HCM Data Loader or HCM Spreadsheet Data Loader. It generates a file of HCM Data Loader DELETE instructions, adds the file to the DeleteExampleObject||<date and time>||.zip file, and uploads it.

**Spreadsheet Business Object Access**

Users who have the predefined Human Capital Management Integration Specialist *job role* can create and maintain spreadsheet templates for any business object. They can also generate spreadsheets from templates and perform all data-set operations, such as create and upload. By default, no other role has this access. This topic describes how to enable other roles to:

- Create and import spreadsheet templates.
- Maintain spreadsheet templates.
- Generate spreadsheets and create, save, and upload data sets from those spreadsheets.
- View all data sets in spreadsheets generated from a specific template.

It also introduces some security setup that you must perform to enable users to access relevant tasks in the Data Exchange work area.

**Creating and Importing Spreadsheet Templates**

To enable users to create and import spreadsheet templates, you perform the **Manage Spreadsheet Business Object Access** task. Using this task, you can:

- Select a role and assign one or more business objects to it.
  
  For example, you can select a custom Compensation Administrator job role and assign all compensation objects to the role. Users who have the job role can create and import spreadsheet templates for all compensation objects.

- Select a single business object and assign one or more custom roles to it.
  
  For example, you can select the Job object and assign custom Human Resource Specialist and Human Capital Management Application Administrator job roles to it. Users who have either role can create and import spreadsheet templates for the Job object.

In both cases, users can’t create and import spreadsheet templates for other business objects, unless they have other custom roles that provide that access.

**Tip:** When you assign business objects individually, you must assign the translation objects explicitly, if required. They’re not assigned by default with the base object.

**Maintaining Spreadsheet Templates**

While spreadsheet templates are in *Draft* status, the users who created them can maintain them. When a spreadsheet template is in other statuses, then only the roles listed in the Template Maintenance section of the spreadsheet template can maintain it.
Generating Spreadsheets and Performing Data Set Operations

Users who can create and import spreadsheet templates determine who can generate spreadsheets and perform data set operations. On the Manage Role Access tab of the spreadsheet template, they identify the roles that can:

- Create, save, and upload data sets from spreadsheets generated from the template.
- View all data sets in spreadsheets generated from the spreadsheet template.

For example, a user with a custom Compensation Administrator job role may create a spreadsheet template named US Compensation Grades for loading Grade objects. On the Manage Role Access tab of the template, that user selects a custom Compensation Analyst role and selects the Create, Save, and Upload options. Therefore, any user with the custom Compensation Analyst job role can:

- Generate spreadsheets for the Grade object from the US Compensation Grades spreadsheet template.
- Create, save, and upload data sets from the generated spreadsheets.

**Note:** The Manage Role Access tab of the spreadsheet template is enabled only if the Assign Roles to Template field is set to Enabled. This field appears in the Template Security section of the Define page of the spreadsheet template.

Enabling Task Access

Users who have the predefined Human Capital Management Integration Specialist job role can access the Data Exchange work area and perform these tasks:

- Manage Spreadsheet Templates
- Run Spreadsheet Data Loader

Other users need access to the same tasks so that they can:

- Create, import, and maintain spreadsheet templates.
- Generate spreadsheets and perform permitted data set operations.

To provide this access, you add relevant privileges to the custom roles that are authorized to use HCM Spreadsheet Data Loader. You must have the IT Security Manager job role or privileges to perform this task.

**Related Topics**

- How You Associate a User Name with a Data Upload

Manage Spreadsheet Business Object Access

Perform the Manage Spreadsheet Business Object Access task to enable selected roles to create and import spreadsheet templates for selected business objects. You can secure this access by either role or business object:

- To secure access by role, you select a single role and assign one or more business objects to it.
- To secure access by business object, you select a single business object and assign one or more roles to it.

This topic describes both methods. In the Setup and Maintenance work area, go to the following for your offering:

- Functional Area: HCM Data Loader
Task: Manage Spreadsheet Business Object Access

Securing Access by Role

To secure access by role, follow these steps:

2. In the Job and Abstract Roles section of the page, enter the name of the role in the Role query-by-example field and press Enter. For example, enter Compensation Administrator Custom.
3. In the results, select the role.

The Assigned Business Objects section title is updated automatically to include the role name.
4. In the Assigned Business Objects section, select an entry in the Assign menu.

If you select Assign Individual Business Objects, then:
   a. Search for and select the business objects in the Search and Select Business Objects dialog box. For example, search by the Compensation product area and select specific objects.
   b. Click Add to add the selected business objects to the role. An entry appears in the Assigned Business Objects section for each selected business object.
5. If you select Assign All Business Objects in a Product Area, then:
   a. Select the product area in the Select Product Area dialog box. For example, select Compensation.
   b. Click Add. A single entry appears for all compensation objects in the Assigned Business Objects section.
6. If you select Assign All Business Objects, then a warning message appears. Click Add to close the warning and continue. A single entry appears for all business objects in the Assigned Business Objects section.
7. Click Save.

Users who have the selected role can now create and import spreadsheet templates for the selected business objects.

Securing Access by Business Object

To secure access by business object, follow these steps:

2. In the Business Object section of the page, enter the name of the business object in the Business Object query-by-example field and press Enter. For example, enter Job.
3. In the results, select the business object.

The Assigned Roles section title is updated automatically to include the object name.
4. In the Assigned Roles section, click the Add icon.
5. In the Search and Select Roles dialog box, search for and select one or more roles. For example, select Human Resource Specialist Custom.
6. Click Add.

An entry appears in the Assigned Roles section for each of the selected roles.
7. Click Save.

Users who have the selected roles can now create and import spreadsheet templates for the selected business object.
How You Enable Access to HCM Spreadsheet Data Loader Tasks

You can perform the following tasks in the Data Exchange work area using the predefined job roles Human Resource Specialist and Human Capital Management Integration Specialist:

- Manage Spreadsheet Templates
- Run Spreadsheet Data Loader
- Delete Spreadsheet Stage Table Data

This topic explains how to give other roles access to these tasks. You must have the IT Security Manager job role or privileges.

Access to the Manage Spreadsheet Templates Task

On the Manage Spreadsheet Business Object Access page, you can authorize selected custom roles to create and import spreadsheet templates for selected business objects. You must also enable those custom roles to perform the Manage Spreadsheet Templates task in the Data Exchange work area. To enable this access, you edit the custom roles to add the following aggregate privileges:

<table>
<thead>
<tr>
<th>Aggregate Privilege Name</th>
<th>Aggregate Privilege Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Data Using HCM Spreadsheet Data Loader</td>
<td>ORA_HRC_LOAD_DATA_USING_HSDL</td>
</tr>
<tr>
<td>Manage HCM Spreadsheet Data Loader Templates</td>
<td>ORA_HRC_MANAGE_HSDL_TEMPLATES</td>
</tr>
</tbody>
</table>

In addition, if the Manage Data Exchange Work Area (HRC_MANAGE_DATA_EXCHANGE_PRIV) function security privilege isn't already granted to the role, then you must grant it.

Access to the Run Spreadsheet Data Loader Task

On the Manage Role Access tab of a spreadsheet template, you can authorize selected custom roles to generate spreadsheets and perform data-set actions. You must also enable those custom roles to perform the Run Spreadsheet Data Loader task in the Data Exchange work area. To enable this access, you edit the custom roles to add the following aggregate privileges:

<table>
<thead>
<tr>
<th>Aggregate Privilege Name</th>
<th>Aggregate Privilege Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access HCM Spreadsheet Data Loader Templates</td>
<td>ORA_HRC_ACCESS_HSDL_TEMPLATES</td>
</tr>
<tr>
<td>Load Data Using HCM Spreadsheet Data Loader</td>
<td>ORA_HRC_LOAD_DATA_USING_HSDL</td>
</tr>
</tbody>
</table>
In addition:

- Roles that are authorized to maintain a spreadsheet template must have the Manage HCM Spreadsheet Data Loader Templates (`ORA_HRC_MANAGE_HSDL_TEMPLATES`) aggregate privilege. You authorize a role to maintain a spreadsheet template on the Manage Role Access tab of the template.
- If the Manage Data Exchange Work Area (`HRC_MANAGE_DATA_EXCHANGE_PRIV`) function security privilege isn’t already granted to the role, then you must grant it.

Access to the Delete Spreadsheet Stage Table Data Task

To enable custom roles to perform the Delete Spreadsheet Stage Table Data task in the Data Exchange work area, add the following function security privilege:

<table>
<thead>
<tr>
<th>Function Security Privilege Name</th>
<th>Function Security Privilege Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete HCM Spreadsheet Data Loader Stage Data</td>
<td>HRC_DELETE_HSDL_STAGE_DATA_PRIV</td>
</tr>
</tbody>
</table>

In addition, if the Manage Data Exchange Work Area (`HRC_MANAGE_DATA_EXCHANGE_PRIV`) function security privilege isn’t already granted to the role, then you must grant it.

Related Topics

- Edit Job and Abstract Roles

FAQs for HCM Spreadsheet Data Loader

Why can't I access HCM Spreadsheet Data Loader?

You may not have the correct permissions. You must have the Human Capital Management Integration Specialist job role or privileges.

In addition, the **HCM Data Loader Scope** configuration parameter must be set to **Full**.

Can I delete data using a spreadsheet?

Yes. In the spreadsheet template, you set the **Supported Action** value to **Delete only**. If the action isn’t available, then the selected object doesn’t support deletion of any of its components. On the Design page of a template for deleting objects, only those components that can be deleted appear.
26  **HCM Spreadsheet Data Loader Templates**

**HCM Spreadsheet Data Loader Templates**

HCM Spreadsheet Data Loader loads HCM business objects from spreadsheets, which you generate from spreadsheet templates. A spreadsheet template for a business object or business object component contains selected attributes in a specified order. Every spreadsheet generated from the template has the same structure and organization as the template. This topic describes spreadsheet templates.

**Spreadsheet Template Features**

You can configure many components of the spreadsheet template. Therefore, you can create spreadsheet templates to suit business requirements. For example, you can:

- Select attributes to include.
- Specify attribute order and column headings.
- Enable and disable attributes for data entry.
- Hide attributes that are included.
- Provide default values.
- Make optional values required.
- Create instruction text for the spreadsheet.
- Configure Lists of Values to provide attribute validation.

**Managing Spreadsheet Templates**

To create a spreadsheet template, you can:

- Copy an existing spreadsheet template and edit the copy.
- Create a spreadsheet template from scratch.
- Import a spreadsheet template that was exported from a different environment or from Oracle Cloud Customer Connect.

**Predefined Spreadsheet Templates**

Predefined spreadsheet templates may exist for commonly loaded objects. These templates contain columns for all supported components and attributes. Search for predefined spreadsheet templates, which have codes beginning **ORA_**, on the Manage Spreadsheet Templates page.

*Tip:  Copy the predefined templates and edit the copies to suit business requirements.*
Spreadsheet Template Status

The status of a spreadsheet template determines whether you can edit it, delete it, or generate spreadsheets from it. This table summarizes the template status values.

<table>
<thead>
<tr>
<th>Spreadsheet Template Status</th>
<th>Edit</th>
<th>Delete</th>
<th>Generate Spreadsheets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Draft</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Active</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inactive</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

⚠️ **Note:** You can disable attributes in an active spreadsheet template but you can’t remove them.

To make an active spreadsheet template unavailable to users, you change its status to **Inactive**. You can’t change it back to **Draft**.

Access to Templates and Generated Spreadsheets

Users who have the predefined Human Capital Management Integration Specialist **job role** can create and maintain spreadsheet templates for any business object. They also perform the **Manage Spreadsheet Business Object Access** task to enable other roles to create and import spreadsheet templates for selected objects.

On the Manage Role Access tab of the template, users who can create and import spreadsheet templates identify custom roles that can:

- Create, save, and upload data sets from spreadsheets generated from the template.
- View all data sets in spreadsheets generated from the template.
- Maintain the template.

⚠️ **Note:** The Manage Role Access tab is enabled in spreadsheet templates for unsecured objects and objects where the application data security is available in HSDL. You can’t manage role access for templates based on secured objects where the data security is not yet available with HCM Spreadsheet Data Loader.

In generated spreadsheets:

- Users see only data sets that they created, unless their role is associated with the spreadsheet template with the **View All** option.
- Users can perform the data-set operations that are enabled for their roles.
- Users with the Human Capital Management Integration Specialist job role can perform all actions and view all data sets.
Translating Spreadsheet Templates

You can provide translated versions of these text entries in a spreadsheet template:

- The template name and description
- Template instructions
- Column headings and descriptions

Set the session language to the target language before editing the spreadsheet template to enter translated text.

Related Topics
- Spreadsheet Business Object Access

Define Categories for Spreadsheet Templates

If you define many spreadsheet templates, then you may want to group them by category to make retrieval easy. No categories are predefined. This topic describes how to define template categories.

Follow these steps:

1. In the Setup and Maintenance work area, go to the Manage Common Lookups task.
2. On the Manage Common Lookups page, search for the lookup type ORA_HRC_DL_TEMPLATE_CATEGORY.
3. In the search results, select ORA_HRC_DL_TEMPLATE_CATEGORY to display its lookup codes.
4. Click the New icon to add a row to the lookup codes table.
5. Define the lookup codes for your categories. For example, to create a category for worker templates:
   a. Enter WORKER_TEMPLATES in the Lookup Code field.
   b. Enter Worker in the Meaning field. This value appears in the list of categories on the Manage Spreadsheet Templates page, for example.

   Tip: The Display Sequence value is a number that specifies the position of this category in the list of categories.

6. Click Save and Close.

You can now select your category on the Manage Spreadsheet Templates and Manage Template Definition pages. You can also search by category when you perform an advanced search for a spreadsheet template on the Load Spreadsheet Data page.

Create and Edit Spreadsheet Templates

Creating a spreadsheet template is a five-step process. You:

1. Define general information for the template.
2. Design the template’s contents and layout.
3. Include template instructions, if required.
4. Specify who can perform data set operations and who can maintain the template.
5. Review the design and generate a test spreadsheet from the template.

This topic provides instructions for the five steps of this process and describes how to edit an existing spreadsheet template.

Define the Template

Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select Manage Spreadsheet Templates to open the Manage Spreadsheet Templates page.
2. Click Create.
3. In the Create Template dialog box:
   a. Enter a name for the template. By default, generated spreadsheets have the name of the template with a date and time stamp.
   b. The Code value is generated automatically based on the Name value. You can overwrite the code.
   c. Select the business object for which you’re creating the template.
   d. Select a legislative data group (LDG), if appropriate. Typically, the LDG is relevant for Global Payroll objects.
   e. Select a value in the Supported Action field. If you select a business object that doesn’t support deletion for any of its components, then Create and update appears by default.
   f. Enter a description and select a category, if required.
   g. Click OK.

The Define page opens. Some fields contain values that you entered in the Create Template dialog box. The template is in draft status by default. Spreadsheet users can generate spreadsheets from active templates only. Leave the template in draft status until it’s ready for use.

4. If the spreadsheet template is for an unsecured object, such as Job, or for an object where data security is supported by HCM Spreadsheet Data loader, then set the Upload Data As value in the Template Security section. This value determines which user name appears in the object’s history. You can’t set this value for secured objects. This step is optional.

5. In the Template Parameters section, set parameter values to apply to all spreadsheets generated from this template. This step is optional.

6. Click the Design tab to open the Design page.

Design the Template

The Available Attributes panel lists all attributes for the top-level component of the object. Use the Available Attributes menu to select a different component. If you selected the Delete only value in the Supported Action field, then only those components that can be deleted appear.

Follow these steps:

1. Add attributes from the Available Attributes panel to the Selected Attributes panel. When you select any attribute, all attributes that are required to identify the object move automatically to the Selected Attributes panel.
2. Either edit individual attributes, as required, or manage all attributes at once by selecting Advanced Options for Attribute Definition.
3. When the template design is complete, click the Add Instructions tab to enter instructions, if required. Otherwise, click the Manage Role Access tab.
4. Reorder the attributes in the Selected Attributes panel to specify the order you require in generated spreadsheets.
Enter Template Instructions

On the Add Instructions page, you can enter instructions to appear on a separate Instructions worksheet in spreadsheets generated from the template. These instructions could describe how loaded objects are to be validated, for example. To enter instructions:

1. Click Add.
2. In the Create Instruction dialog box, enter a heading and the instruction text.
3. Repeat steps 1 and 2 to create additional sections of instruction text, as required. You can reorder and delete sections.

Manage Role Access

On the Manage Role Access tab, you specify which roles can:

- Create, save, or upload data sets created in spreadsheets generated from the template.
- View all data sets created in spreadsheets generated from the template.
- Maintain the template.

Note: Manage Role Access actions are enabled only if the Assign Roles to Template field is set to Enabled. Assign Roles to Template, which is read only, appears in the Template Security section of the Define page. This field is set automatically based on the object for which the template is defined. It’s set to Disabled for secured objects.

Follow these steps:

1. In the Permitted Data Set Operations section of the Manage Role Access page, click the Add icon.
2. In the Search and Select Roles dialog box, search for and select one or more custom roles and click Add.

   The roles appear in the Permitted Data Set Operations section. By default, the selected roles can create and save data sets. You can select the Upload and View All options for any of the roles, if required.

   Tip: The actions that you authorize on the Manage Role Access tab are enabled by default. You can deselect the Enabled option for any role to disable the role’s access temporarily.

3. In the Template Maintenance section of the page, click the Add icon.
4. In the Search and Select Roles dialog box, search for and select one or more custom roles and click Add.

   The roles appear in the Template Maintenance section.

5. Click Save.

Review the Template

On the Review page, your template design appears.

1. To validate the template, click Validate.
2. To generate a sample spreadsheet, click Preview.
Tip: You can click Preview on any page to generate a sample spreadsheet. However, the spreadsheet shows only the configuration that you have saved so far.

3. Either save the file locally or open it. When you open it, you’re prompted to sign in.
4. Review the spreadsheet.
5. Save the template.

When you’re ready to make the template available, edit it and set its status to Active.

Edit the Template

To edit a spreadsheet template, search for it on the Manage Spreadsheet Templates page and click its name in the search results. You can edit an existing spreadsheet template in any status, but you can’t change its code or type. In addition, you can’t remove attributes from an active template, but you can disable them.

Related Topics
- Spreadsheet Business Object Access

How You Associate a User Name with a Data Upload

For some object components, you can review the object history to see who last updated the object. This topic explains how to specify which user name appears in the object history when you upload data using HCM Spreadsheet Data Loader.

Setting Template Security

When you create a spreadsheet template, you select a value for the Upload Data As field in the Template Security section of the Define page. This value determines what appears in an object’s history when data is uploaded from spreadsheets generated from the template.

When you create a spreadsheet template for an object that can be loaded by a business user, such as Job or Grade:

- You can select either Session User or Elevated User in the Upload Data As field. When you select Session User, the user name of the signed-in user who uploaded the data appears in the object history. If you select Elevated User, then the user name is always FUSION_APPS_HCM_ESS_LOADER_APPID. For objects that don’t use data security, or where data security is available for spreadsheet bulk data loading, Upload Data As is set to Session User by default.
- If you set the Upload Data As field to Session user, then you can assign roles to the template. These steps are required to allow business users to upload data with spreadsheets generated from the template.

When you create a spreadsheet template for an object that uses data security in the application, but that data security isn’t available yet for spreadsheet loading, such as Worker:

- Upload Data As is set to Elevated User and can’t be changed. The user name in object history is always FUSION_APPS_HCM_ESS_LOADER_APPID.
• It is not possible to share such templates with other roles. Only users who have the predefined Human Capital Management Integration Specialist job role or privileges can upload data from a spreadsheet for objects that use data security in the application where that data security isn't available for spreadsheet data loading.

Related Topics
• Spreadsheet Business Object Access

Set Spreadsheet Template Parameters

In a spreadsheet template, you can set parameters that control the processing of all spreadsheets generated from the template. Any values that you specify in a spreadsheet template override equivalent enterprise settings. This topic explains how to set spreadsheet template parameters.

Follow these steps:

1. Open the Define page in the Manage Spreadsheet Templates task flow.
2. In the Template Parameters section, click the Add icon.
3. In the Search Template Parameters dialog box, either enter a parameter name or select a category and click Search. This table lists the parameters and their categories.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Category</th>
<th>Description</th>
<th>Enterprise Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Audit Data</td>
<td>Availability</td>
<td>Specifies whether audit data is captured when audit is configured for business objects loaded from generated spreadsheets.</td>
<td>No</td>
</tr>
<tr>
<td>Enable Incremental Load Events</td>
<td>Availability</td>
<td>Enable events to be triggered for the data supplied.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Caution: Enabling load events will significantly increase the time it takes for your data to load.</td>
<td></td>
</tr>
<tr>
<td>Purge Audit Data</td>
<td>File Definition</td>
<td>Specifies whether audit data is purged for objects successfully loaded or updated from the spreadsheet.</td>
<td>No</td>
</tr>
<tr>
<td>Maximum Concurrent Threads for Load</td>
<td>Performance</td>
<td>The maximum number of threads to be used concurrently to load spreadsheet data.</td>
<td>8</td>
</tr>
<tr>
<td>Calculate Worker Full-Time Equivalent</td>
<td>Scheduling Default</td>
<td>Specifies whether to calculate automatically the full-time equivalent value of the working hours on a worker assignment.</td>
<td>No</td>
</tr>
</tbody>
</table>
### HCM Spreadsheet Data Loader Templates

**Parameter** | **Category** | **Description** | **Enterprise Default Value**
--- | --- | --- | ---
Create Worker Default Work Hour Pattern | Scheduling Default | Specifies whether to create default working hour patterns automatically for workers loaded without working hour patterns. | No

- This parameter applies to templates for the Worker object only.

Date-Effective Update Mode | Scheduling Default | When set to Replace, specifies that the data in spreadsheets replaces any existing data in Oracle HCM Cloud. When set to Retain, specifies that any existing future-dated changes are retained. | Retain

- This parameter applies to templates for the Worker object only.

Initiate Business Object Post Processing | Scheduling Default | Specifies whether to run processes that are configured to run automatically after the object is loaded. | Yes

Maximum Percentage of Load Errors | Scheduling Default | The percentage of business object instances in error that can occur for a business object before the validation and load processes stop. | 100

---

4. In the search results, select a parameter and click **OK**. The selected parameter appears in the Template Parameters section.
5. Enter or select a value for the parameter, as appropriate.
6. Repeat from step 2 for additional parameters.
7. Either click **Save and Close** or select a different page in the task flow.

You can set parameters to support each template’s use case. For example, you can set **Date-Effective Update Mode** to:

- **Replace**, in a spreadsheet template for loading new hires
- **Retain**, in a spreadsheet template for updating worker records

**Tip:** You can set spreadsheet parameter values for the enterprise on the HCM Spreadsheet Data Loader Parameters tab of the Configure HCM Data Loader page.

**Related Topics**

- HCM Data Loader Configuration Parameters
Spreadsheet Template Design

Creating a spreadsheet template is a five-step process. You:

1. Define general information for the template.
2. Design the template’s contents and layout.
3. Specify instruction text, if required.
4. Identify the roles that can perform data set operations, such as create and upload, and the roles that can maintain the template.
5. Review the design and generate a test spreadsheet from the template.

This topic describes the design step of this process.

Available Attributes Panel

The Available Attributes panel lists all attributes of the selected component of the object. You can:

- Set the Show field to filter the attributes.
- Select a different component or all components.
- Switch to a tree view to see the component hierarchy. For each component of the object, you see separate entries in the hierarchy for attributes, flexfield attributes, and child components.
- Click an attribute name to view the attribute details.
- Move attributes from the Available Attributes panel to the Selected Attributes panel. When you move the first attribute, any required attributes for the component move automatically to the Selected Attributes panel. Attributes that are required conditionally don’t move. You must add them explicitly to include them.

Note: On occasion, you may see two attributes with the same name in the Available Attributes panel. Typically, this duplication occurs for foreign object references. It enables you to choose between entering free text and selecting a value from an LOV. The icons that precede each name show the attribute type.

Selected Attributes Panel

In the Selected Attributes panel, attributes appear in the order in which you added them. You can:

- Edit the details of an attribute. For example, you can:
  - Change the column heading.
  - Make an optional value required.
  - Specify a default value. You can also make attributes, such as those with default values, read-only.
  - Specify whether the attribute is visible in the generated spreadsheet.
  - Disable optional attributes. These attributes appear but can’t be updated.
  - Enter a description to appear as help text in the generated spreadsheet.
  - Configure a list of values to validate the attribute.

- Create Note attributes. Use Note attributes to return values from a list of values for information only. For example, you may have a list of values for the PersonNumber attribute that returns the person number to this attribute. It can
also return the person’s last and first names in the **Note** column. You can make the **Note** column read-only, and multiple **Note** columns can have values returned from the same list of values. Data in **Note** columns isn’t passed to the application, but can be used in the data set for information purposes or coordination of multiple contributors.

You can also use **Note** attributes to communicate information that doesn’t need to be uploaded. For example, the Human Resources Specialist may have submitted overtime information in a spreadsheet for the Payroll Administrator to action.

- Change the attribute order. After you determine if any attributes are hidden, click the **Reorder** icon to open the Reorder Attributes dialog box.
- Duplicate a component. This option exists for any component that can appear more than once in an object, such as Person Phone or Location Legislative Extra Information.

**Tip:** Edit the selected attributes of the component before creating additional instances.

### Advanced Settings

To manage all attributes at once, click the **Advanced Options for Attribute Definition** icon. On the Advanced Options for Attribute Definition page, you can manage:

- Basic information, flexfield attributes, and lists of values.
- Attribute keys. When you import data to a spreadsheet from a file, you can include a header. The attribute names used in the import file can be different from those in the spreadsheet. In this case, the **Attribute Key** fields must include the names from the import file. For example, you can import a location code to a spreadsheet from a file where it’s known as the location ID. To ensure that the data imports successfully, you must set the **Attribute Key** value to **Location ID** for the **Location Code** attribute.

### Guidelines for Designing Spreadsheet Templates

Spreadsheet templates are highly configurable. Therefore, you can design them to support specific business use cases. This topic provides some guidelines to help you create spreadsheet templates that minimize the scope for error and are easy to use.

### Supported Actions

Depending on the business object, you can use HCM Spreadsheet Data Loader to either create and update or delete object components. You can design a single spreadsheet for both creating and updating an object. However, spreadsheet templates that support a single action are better for users and easier for you to manage. For example:

- In templates for creating objects, you can mark as required all attributes that must be supplied for new objects. Typically, the same attributes aren’t required when updating objects, so you can mark different attributes as required in templates for updating objects.

  **Tip:** To mark an attribute as required, edit the attribute in the Selected Attributes panel on the Design page and select **Required**.

- In templates for updating objects, you can configure lists of values to identify records to update.
Lists of Values
For many attributes, you can ensure that users enter only valid data by providing choice lists and searchable lists of values. Some attributes have lists of values by default. In particular:

- Some lookup-validated attributes are automatically configured with choice lists. A list of values is displayed when the user clicks in the cell.
- To reference a foreign object from your template, you can include the Oracle Fusion Surrogate ID attribute. Attributes like this are automatically configured with searchable lists of values, and the selected attribute is returned to the spreadsheet. While a surrogate ID itself may not be meaningful, you can also configure your own lists of values for attributes included in a spreadsheet. These attributes can be validated using a lookup type, value set, or view object.

For example, you can return additional values to other attributes, such as Person Number, instead of just returning the person ID. You can also copy the list of values configuration to another attribute that is typically used to identify a record, such as the Name or Number identifier. You configure the list of values to return the recognizable number.

Tip: Attributes with lists of values have the List of values icon before their names in the Available Attributes panel of the spreadsheet template.

Default Values
You can configure default values for attributes in spreadsheet templates. You may want to exclude attributes with default values so that you can enforce their values. Alternatively, you can include the attribute so that users can enter a value, which overrides the default value.

Date Editors
A date editor is provided automatically for all date attributes. However, the user must click in the spreadsheet row to initialize date editors. Therefore, you’re recommended not to make a date value the first column in the row. This approach ensures that date editors in the row are initialized before they’re used.

How You Configure Default Values for Spreadsheet Attributes
When you design a spreadsheet template for bulk data loading using HCM Spreadsheet Data Loader, you can specify default values for attributes. This topic describes the ways in which you can specify default values.

Specifying Default Values for Date Attributes
This table describes the options for specifying default date values.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specified date</td>
<td>Using the date editor, select a date.</td>
</tr>
</tbody>
</table>
Option | Description
--- | ---
Load date | Use the date when the spreadsheet row is loaded.
Existing selected attribute | Select another date attribute to supply the default value. The attribute must be in the Selected Attributes section of the Design page.
Use expression | Supply a Groovy expression to derive the default value.

For example, to set the default value of a date attribute to the load date:

1. Select the attribute in the Selected Attributes section of the Design page of Manage Spreadsheet Templates and click **Edit**.
2. In the Edit dialog box, set **Default Value** to **Value** and select **Load date** from the list.

**Specifying Default Values for Attributes Other Than Dates**

This table describes the options for specifying default values for attributes other than dates.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant value</td>
<td>Enter the default value.</td>
</tr>
<tr>
<td>Existing selected attribute</td>
<td>Select another attribute of the same data type to supply the default value. The attribute must be in the Selected Attributes section of the Design page.</td>
</tr>
<tr>
<td>Use expression</td>
<td>Supply a Groovy expression to derive the default value.</td>
</tr>
</tbody>
</table>

**Supplying Groovy Expressions**

Using Groovy expressions enables you to manipulate other attribute values to provide a default value. To specify a Groovy expression to derive a default value:

1. Select the attribute in the Selected Attributes section of the Design page and click **Edit**.
2. In the Edit dialog box, set **Default Value** to **Value** and select **Use expression** from the list.
3. Enter the expression in the text area and click **Validate** to ensure the expression is valid.

**Hiding Attributes**

You may want to hide attributes that have default values so that spreadsheet users aren’t aware of them. To hide an attribute, edit it in the Selected Attributes panel on the Design page of the template. In the Edit dialog box, deselect the **Visible** option.
Configure Lists of Values for Attribute Validation

Some types of attributes are configured automatically with choice lists or searchable lists of values in spreadsheets. For such attributes, spreadsheet users can select only valid values, which simplifies data loading. You can configure lists of values for other attributes to make populating spreadsheets easier. This topic describes how to configure lists of values.

Select the LOV Source for an Attribute

To validate an attribute using a list of values, you must identify the list source. It can be one of:

- Lookup
- Value set
- View object

Select View object to include user key values in a spreadsheet in place of a surrogate ID to reference a foreign object. Having user keys helps the spreadsheet user, who sees recognizable values in the spreadsheet. For example, in a spreadsheet to create positions you include a reference to a department. You could just include the Department ID attribute in the spreadsheet, as it’s automatically configured with a searchable list of values. However, when the user selects a value from the list, the surrogate ID is returned to the spreadsheet. Instead, you could include the Department user key attribute in the spreadsheet. Using a view object, you configure the Department attribute with a searchable list of values to return the department name to the spreadsheet.

Validate Attributes Using Lookups or Value Sets

Follow these steps:

1. On the Design page of the Manage Spreadsheet Templates task flow, move attributes to the Selected Attributes pane.
2. In the Selected Attributes pane, select the attribute to be validated with a lookup type or value set and click Edit.
3. Complete the details on the main tab of the Edit dialog box, as appropriate.
   - When you’re editing the details of an existing list of values, you can change the list display type on the main tab, if required.
4. Click the List of Values tab.
5. Set the LOV Source value to Lookup or Value set, as appropriate.
6. Select the name of the lookup type or value set.
7. Specify whether the LOV is to appear as a choice list or a searchable list of values in generated spreadsheets.

   **Tip:** If the list of values has more than 20 entries, then a searchable list of values is better for the spreadsheet user. Use choice lists only when the number of choices is limited.

8. Select attributes to appear in the LOV.
9. Identify the values to be returned to the spreadsheet from the choice list or search.

If the LOV is a searchable list of values, then you can add filter and search criteria.

- Add filter criteria to enable the user to filter the returned list of attribute values.
- Add search criteria to enable the user to search the returned list of attribute values.
To add filter criteria, in the Filter Criteria section:

1. Select a value in the Filter Criteria field and click Add. The View Criteria and Bind Variable values are completed automatically.
2. Set the Expression Type value for each of the view criteria to one of the following values:
   - Constant
   - Expression
   - Attribute
3. If the Expression Type is set to Constant or Expression, then enter the constant or expression in the Value field.
   If the Expression Type is set to Attribute, then select the attribute in the Value field.

To add search criteria, in the Search Criteria section:

1. Select a value in the Search Criteria field and click Add. The Search Criteria and Attribute fields are completed automatically.
2. Repeat step 1 for additional search criteria.

Click OK to close the Edit dialog box.

Validate Attributes Using View Objects

Follow these steps:

1. On the Design tab of the Manage Spreadsheet Templates task flow, move both the surrogate ID and user key attributes to the Selected Attributes pane.
2. Edit the surrogate ID attribute. On the List of Values tab in the Edit dialog box, the LOV Source field shows the full path to the view object. For example, for the Department ID attribute you would see the following value:
   oracle.apps.hcm.organizations.publicView.DepartmentPVO
   Copy this path and close the Edit dialog box.
3. Edit the primary user-key attribute and set LOV Source to View object.
4. In the LOV Source field, paste the full path to the view object that you copied from the surrogate ID attribute.
5. Leave Displayed As set to Searchable list of values.
6. Select the attributes to appear in the searchable list.
7. Identify the attribute values to be returned to the spreadsheet from the search. Usually, you include both the surrogate ID and one or more user key values. The user key values are for the spreadsheet user, and the surrogate ID is used to load the data. You can hide the surrogate ID attribute in generated spreadsheets by deselecting the Visible option in the Edit dialog box for the attribute.
8. Enter filter and search criteria, as required.
9. Click OK to close the Edit dialog box.

Autogenerated Attribute Values in Spreadsheet Templates

In spreadsheet templates for the Worker object you can specify that values for some attributes, such as Person Number and Assignment Number, are generated automatically. For example, on the Design page in a Worker spreadsheet template, you edit the Person Number attribute in the Selected Attributes panel and select Autogenerate. In this case, users don’t have to supply the person number value in generated spreadsheets. However, so that related records can be identified and processed together, a unique identifier must still be supplied. Therefore, Person Number must appear
in generated spreadsheets and its value must be a unique identifier. This value isn’t used as the person number, which is generated on successful creation of the worker record.

To ensure that users supply this unique identifier for a logical object, these rules apply to the template definition:

- If you select the Autogenerate option for an attribute, then the Required option is selected automatically and made read-only.
- If the Required option is selected and the Default Value field is set to None, then the Visible option is selected automatically and made read-only.
- A default column heading is defined. For example, for the Person Number attribute the column heading is Person Unique Identifier. For the Assignment Number attribute, the column heading is Assignment Unique Identifier. You can change the default column heading.
- The default description of the attribute explains that a unique identifier must be supplied. You can edit this description.

Spreadsheets generated from the template include the Person Number or Assignment Number attribute, as appropriate, with its new column heading and description. For the attribute value, users must enter a unique identifier to ensure that records are loaded successfully.

Copy Spreadsheet Templates

Copying a spreadsheet template and editing the copy is an efficient way to create a new spreadsheet template. This topic describes how to copy spreadsheet templates. You perform this task in the Data Exchange work area.

Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select Manage Spreadsheet Templates.
2. On the Manage Spreadsheet Templates page, search for the spreadsheet template to copy.
3. Select the spreadsheet template in the search results and select Copy from the template Actions menu.
4. In the Copy Template dialog box:
   a. Enter a unique name for the spreadsheet template in the Name field.
   b. A Code value is created automatically based on the name value. You can overwrite the code value.
   c. Select a legislative data group, if the template is legislation-specific.
   d. Enter a description and select a category, if required.
   e. Click OK.
5. Click OK to close the Confirmation dialog box.
6. On the Manage Spreadsheet Templates page, search for the new spreadsheet template.
7. In the search results, click the name of the spreadsheet template to open it for editing.

You can copy a spreadsheet template in any status, but the status of the new spreadsheet template is always Draft. Change the status to Active when the template is ready to use. Active spreadsheet templates are available on the Run Spreadsheet Data Loader page.
Export and Import Spreadsheet Templates

You can import spreadsheet templates to other environments to avoid having to recreate them. For example, you can export spreadsheet templates from a test environment and import them to a production environment. This topic describes how to export and import spreadsheet templates. You perform these tasks in the Data Exchange work area.

Export Spreadsheet Templates

Follow these steps in the source environment:

1. On the Tasks panel tab of the Data Exchange work area, select Manage Spreadsheet Templates.
2. On the Manage Spreadsheet Templates page, search for and select the spreadsheet template that you want to export.
3. From the Actions menu for the spreadsheet, select Export.
4. In the Opening dialog box, select Save file and click OK.
5. In the file-name dialog box, select a location for the file and click Save.

The file is exported to the specified location in XML format. Deliver the file to the target environment.

Import Spreadsheet Templates

Follow these steps in the target environment:

1. On the Tasks panel tab of the Data Exchange work area, select Manage Spreadsheet Templates.
2. On the Manage Spreadsheet Templates page, click Import.
3. In the Import dialog box:
   a. Enter a unique name for the spreadsheet template. This name applies in the target environment.
   b. The Code field is completed automatically based on the name value. You can overwrite the value.
   c. If your template is legislation-specific, then select the legislative data group.
   d. Click Choose File to select the template file.
   e. Enter a description and select a category, if required.
   f. Click OK in the Import dialog box to import the template.

Click OK to close the Confirmation dialog box.
4. On the Manage Spreadsheet Templates page, search for the imported template.

The status of an imported spreadsheet template is always Draft. Change the status to Active when the template is ready for use.

How You Enable Business Users with Custom Roles to Load Spreadsheet Data

Business users can load data for objects such as jobs and locations from HCM Spreadsheet Data Loader spreadsheets in their work areas. These business users can generate spreadsheets and load data from them because they have predefined
roles that provide the access they need. This topic explains how to give the same access to business users with custom roles.

For example, let’s say that your human resource specialists create job definitions in bulk. They load them from a spreadsheet on the Manage Jobs page in the Workforce Structures work area. They can generate the spreadsheet and load the data because:

- They have the predefined Human Resource Specialist job role. This job role inherits the Workforce Structures Management duty role, which inherits the Load Data Using HCM Spreadsheet Data Loader aggregate privilege.
- The Job (ORA_PER_JOB) spreadsheet template behind the Manage Jobs page lists the predefined Human Resource Specialist job role on its Manage Role Access page. Users with the role can create, save, and upload data.

Human resource specialists who use a custom role can’t load data from a spreadsheet by default. To let them load data, you:

- Add the Load Data Using HCM Spreadsheet Data Loader aggregate privilege to their custom role. You need the IT Security Manager job role or privileges to update custom roles.
- List the custom role on the Manage Role Access page in the predefined spreadsheet template for the object. You need the Human Capital Management Integration Specialist job role or privileges to update predefined templates.

You can make these changes for custom versions of any predefined role that can load data from HCM Spreadsheet Data Loader spreadsheets.

**Related Topics**

- Edit Job and Abstract Roles

### Add a Custom Role to a Predefined Spreadsheet Template

Your business users may need to load data in bulk from HCM Spreadsheet Data Loader spreadsheets in their work areas. If they’re using custom roles, then you need to give those roles permissions to generate and use the spreadsheets. This topic describes how to add custom roles to predefined spreadsheet templates.

Suppose you want users with the Human Resource Specialist Custom role to load grades on the Manage Grades page in the Workforce Structures work area. Follow these steps:

1. Sign in with the Human Capital Management Integration Specialist job role or privileges.
2. Select **Navigator > My Client Groups > Data Exchange**
3. In the Data Exchange work area, select the **Manage Spreadsheet Templates** task on the Tasks panel tab.
4. On the Manage Spreadsheet Templates page, search for the **Grade** (ORA_PER_GRADE) spreadsheet template.

⚠️ Tip: The codes of predefined spreadsheet templates begin with the characters ORA_.

5. Click the template name in the search results to open the spreadsheet template for editing.
6. Click the Manage Role Access tab in the spreadsheet template.
7. Click the **Add** icon in the Permitted Data Set Operations section of the Manage Role Access page.
8. Select the **Human Resource Specialist Custom** role in the Search and Select Roles dialog box and click **Add**. Your custom role appears in the Permitted Data Set Operations section. The **Create** and **Save** options are selected automatically. They mean that users with this role can create data sets in spreadsheets generated from the template and save them.
9. Select **Upload** to let users upload their data. Select **View All** to let users view spreadsheets generated by other users from the template.
10. Click **Save** then **Done**.

**Tip:** You can't change the attributes in a predefined spreadsheet template, but you can set the template parameters and manage role access.

**FAQs for HCM Spreadsheet Data Loader Templates**

**Can I hide attributes in spreadsheet templates?**

Yes. In the Selected Attributes panel on the Design tab of the template, edit the attribute. In the Edit dialog box, deselect the **Visible** option.

**Why can't I manage role access for some objects?**

For secured objects where data security is not yet available for spreadsheet data loading you can't manage role access in the spreadsheet template. You can manage role access only for unsecured objects and objects where data security is implemented for spreadsheet bulk loading.

Check the read-only **Assign Roles to Template** field on the Define page of the template. If it's set to **Disabled**, then you can't manage role access for the object.

**Can I load data from a spreadsheet as the session user?**

Yes. In the spreadsheet template from which you generate the spreadsheet, **Upload Data As** must be set to **Session User**. This setting is for unsecured objects and those secured objects where data security is implemented for spreadsheet data loading.
Generate Spreadsheets

To load data using HCM Spreadsheet Data Loader, you must first generate a spreadsheet from a spreadsheet template. This topic describes how to generate a spreadsheet and save it locally. You perform this task in the Data Exchange work area, or directly in the page that maintains the object, such as Jobs.

Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select **Run Spreadsheet Data Loader**.
2. On the Run Spreadsheet Data Loader page, the saved search runs automatically. If the template that you want to use doesn’t appear, then click **Show Filters**, enter search terms for the template, and click **Search**.

  **Tip:** In the search results, click the **Template Overview** icon for a template. The Template Overview dialog box provides a list of the attributes that the template supports.

3. Click the name of the template that you want to use.
4. In the Opening GenericHdlSpreadsheet.xlsx dialog box, you will be prompted to log in. You should log in before saving the file, otherwise the **Save File** won’t have the columns generated for the template shape.
5. In the file-name dialog box, select a location and save the spreadsheet.

You can now enter data in the spreadsheet and upload it.

Related Topics

- HCM Spreadsheet Data Loader Templates

Import Data to a Spreadsheet

To populate a spreadsheet with data for upload, you can either enter the data manually or import it from a file. This topic describes how to import data to a spreadsheet from a file.

Follow these steps:

1. If the spreadsheet isn’t already open, then open it and sign in when prompted.
2. You import data to a data set. You can either create the data set or use an existing data set.
   - To create a data set, click **Create Data Set** on the Spreadsheet Loader toolbar. A name is assigned automatically to this new data set. You can overwrite the data set name.
   - To use an existing data set, select its name in the Search section of the spreadsheet and select a **Search Mode** value. Click **Search** in the Spreadsheet Loader toolbar.
3. On the Spreadsheet Loader toolbar, click **Import File**. The Import from File dialog box opens.
4. In the Import from File dialog box, select one of **File Type** values shown in this table.
File Type | Description
---|---
Comma separated values | Comma-separated (.csv) file
Pipe separated values | Delimited data (.dat) file
Other file type | Text file (.txt) with a user-specified separator
XML file type | XML file

**Tip:** You can download CSV and XML file templates from a spreadsheet template. Use them to load data to a spreadsheet generated from the same template.

5. For file types other than **XML file type**, indicate whether the file contains a header.
   
The header provides the names of the attributes for which the file contains values. In this case, the attribute order in the file doesn’t have to match the spreadsheet column order. If you deselect **Header included**, then the order of the values in the file must match the order of the columns in the spreadsheet.
   
In an XML file, the XML tags used for the attribute values must use the attribute keys configured in the spreadsheet template.

6. Click **Browse** to select the file to import.

7. For file types other than **XML file type**, expand the Advanced File Options section. Complete the options as shown in this table for the selected file type. You can use the default values where supplied.

<table>
<thead>
<tr>
<th>File Type</th>
<th>File Separator</th>
<th>Escape Indicator</th>
<th>Newline Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comma separated values</td>
<td>Not applicable</td>
<td>1 - 10 characters</td>
<td>1 - 10 characters</td>
</tr>
<tr>
<td>Pipe separated values</td>
<td>Not applicable</td>
<td>1 - 10 characters</td>
<td>1 - 10 characters</td>
</tr>
<tr>
<td>Other file type</td>
<td>1 - 10 characters</td>
<td>1 - 10 characters</td>
<td>1 - 10 characters</td>
</tr>
</tbody>
</table>

The **Newline Command** field shows the newline command constructed from the supplied **Escape Indicator** and **Newline Indicator** values.

8. Click **OK** to import the data from the selected file. After the data is imported, you can edit any of the imported attribute values.

**Note:** Some validation of the imported data occurs. Any errors appear automatically in the **Progress** and **Message** columns of the **Spreadsheet Line Status** section of the spreadsheet. Correct any errors before you attempt to upload the data.
Manage Spreadsheet Data Sets

A data set is all data loaded from an instance of the same spreadsheet and identified by the same data-set name. You can load the data from a spreadsheet all at once. Alternatively, you can add data rows to an existing spreadsheet and upload them to the existing data set. This topic describes how to create and manage spreadsheet data sets.

Create a Data Set

Follow these steps:

1. Open the generated spreadsheet.
2. In the Connect dialog box, click Yes and sign in when prompted. After a few seconds, the spreadsheet shows all attributes from the selected spreadsheet template.
3. Click the Spreadsheet Loader tab in the spreadsheet toolbar.
4. Click Create Data Set and click OK to close the dialog box. A data set name is generated automatically, based on the template name, date, and time, but you can overwrite it. The name must be unique for the spreadsheet template.
5. Click OK to close the confirmation dialog box.

You can either enter data manually in the spreadsheet or import it from a file.

Update a Data Set

To retrieve an existing data set, follow these steps:

1. Either open the original spreadsheet or generate a new spreadsheet from the same template as the original spreadsheet.
2. In the Data Set Name field under the Search header, select the name of the data set that you want to update. Available data sets:
   - Were uploaded from a spreadsheet that was generated from the same spreadsheet template as the current spreadsheet.
   - Still exist in the stage tables. You can’t update a data set after you have deleted it from the stage tables.
3. Select a Search Mode value. You can retrieve either all records in the data set or only those that failed to load.
4. In the Spreadsheet Loader toolbar, click Search. The specified data-set records are downloaded to the spreadsheet.

You can correct failed rows and add new rows, but you can’t update any row that loaded successfully. Rows that you add belong to the data set that you’re updating. When you upload the data, only new and updated rows are imported to the stage tables.

Save a Data Set

You can import a data set periodically to the HCM Data Loader stage tables by clicking Save on the Spreadsheet Loader toolbar. The data isn’t loaded to the application tables until you click Upload. The Save action lets you:

- Import data to the stage tables over a period of time. For example, you can enter and save payroll adjustments throughout the week but upload them only at the end of the week.
- Create large volumes of data that you save regularly to a single data set. You don’t have to manage those volumes all at once in the spreadsheet.
• Allow multiple users to supply data for processing in the same data set.

Related Topics
• Maintain the HCM Spreadsheet Data Loader Stage Tables

Upload Data Using HCM Spreadsheet Data Loader

In this example, you learn how to upload jobs for the Sales Department using HCM Spreadsheet Data Loader. This table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which business object am I loading?</td>
<td>Job</td>
</tr>
<tr>
<td>Which spreadsheet template am I using?</td>
<td>Sales Jobs</td>
</tr>
<tr>
<td>How do I populate the spreadsheet?</td>
<td>Import the file Sales_Jobs.csv which is a comma-separated file that contains a header and uses the \n newline command.</td>
</tr>
<tr>
<td>Will I create a data set or import the file to an existing data set?</td>
<td>Create a data set.</td>
</tr>
</tbody>
</table>

Summary of the Tasks

Upload the jobs by:

1. Generating the spreadsheet
2. Importing Job objects to the spreadsheet
3. Uploading jobs and correcting errors

Prerequisites

Before you can complete this task:

1. You must have installed the desktop client Oracle ADF 11g Desktop Integration Add-in for Excel.
2. You must have enabled the Trust Center setting **Trust access to the VBA project object model** in Microsoft Excel.
3. The Sales Jobs spreadsheet template must exist and be active.
4. The Sales_Jobs.csv file must exist.
Generating the Spreadsheet

1. On the Tasks panel tab of the Data Exchange work area, select **Run Spreadsheet Data Loader**.
2. On the Run Spreadsheet Data Loader page, click **Show Filters**.
3. Enter **Sales Jobs** in the **Name** field and click **Search**
4. In the search results, click the name of the **Sales Jobs** template.
5. In the spreadsheet dialog box, select **Save File** and click **OK**.
6. In the file-name dialog box, select a location. Enter **SalesJobs.xlsx** in the **File name** field and click **Save**.

Importing Job Objects to the Spreadsheet

1. Open the **SalesJobs.xlsx** spreadsheet and sign in when prompted.
2. On the Spreadsheet Loader toolbar, click **Create Data Set**. Click **OK** to close the warning message, and click **OK** to close the confirmation message.
   The data set is created with a generated name.
3. On the Spreadsheet Loader toolbar, click **Import File**.
4. Complete the fields of the Import from File dialog box as shown in this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Type</td>
<td>Comma separated values</td>
</tr>
<tr>
<td>Header Included</td>
<td>Yes</td>
</tr>
<tr>
<td>File</td>
<td>Sales_Jobs.csv</td>
</tr>
<tr>
<td>Escape Indicator</td>
<td>\</td>
</tr>
<tr>
<td>Newline Indicator</td>
<td>n</td>
</tr>
</tbody>
</table>

5. Click **OK**.
6. Correct any Import File Errors reported in the **Progress** and **Message** columns of the Spreadsheet Line Status section of the spreadsheet.

Uploading Jobs and Correcting Errors

1. On the Spreadsheet Loader toolbar, click **Upload**.
   Data is imported to the stage tables. Objects that import successfully are loaded automatically to the application tables.
2. Click **Refresh**.
   Errors from both import and load are reported automatically in the **Progress** and **Message** columns of the Spreadsheet Line Status section of the spreadsheet.
3. Correct any errors in the spreadsheet.
4. Repeat from step 1 of Uploading Jobs and Correcting Errors until all rows are imported and loaded successfully.
5. Close the spreadsheet.

Related Topics

- Set Up Desktop Integration for Excel

How You Delete Data Using HCM Spreadsheet Data Loader

You can delete business objects using HCM Spreadsheet Data Loader, regardless of how they were created. This topic describes how to delete data using HCM Spreadsheet Data Loader.

Designing the Spreadsheet Template

To delete data, you create a spreadsheet template and set the Supported Action value to Delete only. You can use spreadsheets generated from that template to delete specified objects. The Delete only option for the Supported Action field is available only for objects with at least one component that can be deleted. The object components that appear on the Design tab of the Manage Spreadsheet Templates page are those that you can delete. For example, for the Job object you see the Job Evaluation, Job Valid Grade, Job Extra Information, and Job Legislative Extra Information components. You don't see the parent Job component, because you can't delete it.

When you move an attribute to the Selected Attributes panel of the Design page, all attributes required for deleting the component move across simultaneously.

Copying Spreadsheet Templates

When you copy a spreadsheet template, you can't change the supported action. Therefore, if the source template has Delete only as its supported action, then the copy has the same supported action.

Uploading the Spreadsheet

To delete object components using a spreadsheet, you enter attribute values to identify the components and upload the spreadsheet. When you delete a parent component, its child components are deleted automatically.

Tip: You can monitor progress on the Import and Load Data page. Search for data sets by file type to list those loaded from spreadsheets.

FAQs for Importing and Loading Data Using HCM Spreadsheet Data Loader
How do I fix HCM Spreadsheet Data Loader errors?

You fix errors in the spreadsheet from which you uploaded the data. Errors are reported in the Progress and Message columns of the Spreadsheet Line Status section of the spreadsheet when you click Refresh. After correcting any errors, click Upload again.

Can I load date-effective data?

Yes. You can create, update, and correct date-effective objects. To load date-effective history, you provide a separate row for each date range.

Can I change first and last effective dates?

Yes. Include the Replace First Effective Start Date and Replace Last Effective End Date attributes in the spreadsheet template, as appropriate. Spreadsheet users set these attributes to Y and provide revised dates on the Effective Start Date and Effective End Date attributes.

For some objects, you can change only one of these dates.

Can I load translated objects?

Yes. You don’t have to create a separate spreadsheet template for the translation object. Instead, include translation attributes, such as Language, in the spreadsheet template for the base object. You can use spreadsheets generated from the template to maintain both base-object and translation data.
Chapter 28

Postload Processing

Processes to Run After Loading Data

After a successful data migration or incremental update, you must run processes to complete the setup of some business objects. This topic describes these processes.

This table identifies processes to run after loading Worker objects. The table also suggests the order in which to run the processes when you’re running them all manually. By default, these two processes run automatically when a load of Worker objects completes:

- Refresh Manager Hierarchy
- Update Person Search Keywords

You can prevent either or both of these processes from running automatically using a SET instruction in the Worker.dat file. Alternatively, you can prevent the processes from running for the enterprise by setting configuration parameters.

<table>
<thead>
<tr>
<th>Process</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronize Person Records</td>
<td>1</td>
</tr>
<tr>
<td>Refresh Manager Hierarchy</td>
<td>2</td>
</tr>
<tr>
<td>Update Person Search Keywords</td>
<td>3</td>
</tr>
<tr>
<td>Optimize Person Search Keywords Index</td>
<td>4</td>
</tr>
<tr>
<td>Autoprovision Roles for All Users</td>
<td>5</td>
</tr>
<tr>
<td>Send Pending LDAP Requests</td>
<td>6</td>
</tr>
<tr>
<td>Send Personal Data for Multiple Users to LDAP</td>
<td>7</td>
</tr>
<tr>
<td>Apply Name Formats to Person Names</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

Run these processes in the Scheduled Processes work area. Select Navigator > Tools > Scheduled Processes.

Process Descriptions

**Apply Name Formats to Person Names**

Applies the Display Name, List Name, Full Name, or Order Name format to denormalized person-name data when you add or update a name format. Optionally, you can run the process for a single country.
Autoprovion Roles for All Users
Generates requests to provision and remove roles based on current role-provisioning rules.

Optimize Person Search Keywords Index
Optimizes the index of the PER_KEYWORDS table to improve search performance. By default, the process runs for a maximum of 180 minutes and performs a full optimization. You can enter a different Maximum Optimization Time. If you select the Rebuild the index option, then the Maximum Optimization Time value is ignored. Rebuilding the index can be time consuming.

Run this process whenever you run Update Person Search Keywords. Run it after that process completes.

Refresh Manager Hierarchy
For performance reasons, the complete manager hierarchy for each person is extracted from active data tables. The hierarchy is stored in a separate manager hierarchy table, known as the denormalized manager hierarchy (PER_MANAGER_HRCHY_DN). This approach ensures that a person’s manager hierarchy is both easily accessible and up to date. The Refresh Manager Hierarchy process populates the denormalized manager hierarchy table with latest information after each data load.

If this process isn’t enabled to run automatically, then run it manually after all data loads.

Send Pending LDAP Requests
Sends user-account and role-provisioning requests to your LDAP directory.

Send Personal Data for Multiple Users to LDAP
Ensures that personal data held in your LDAP directory matches that held by Oracle HCM Cloud.

Synchronize Person Records
Notifies consuming Oracle Cloud applications, such as Oracle Fusion Trading Community Model, of changes to person and assignment details since the last data load.

For a once-only load, submit this process with the After Batch Load parameter set to Yes. In the From Date field, enter the start date of the load. In the To Date field, enter the end date of the load. For ongoing updates with daily changes to person records, schedule the process without specifying start and end dates.

Update Person Search Keywords
Several attributes of person, employment, and profile records are used as person-search keywords. This process copies keyword values in all installed languages from the originating records to the PER_KEYWORDS table, where they’re indexed to improve search performance.

Select a person name in the Name field to run the process for a single worker. If you leave the field blank, then the process runs for all workers, which can be time consuming. To avoid performance problems, submit the process with the After Batch Load parameter set to Yes. In this case, the process creates keywords for new records but doesn’t update those for existing records. Leave the Batch ID field blank.
Actions to Perform After Loading Trees

After loading department or organization tree nodes, you must activate your tree version. For more information, see the Implementing Global Human Resources guide.

Related Topics

- Why You Run the Send Pending LDAP Requests Process
- Autoprovisioning
- The SET Instruction

Guidelines for Running Postload Processes

This topic provides additional guidance for some postload processes. In particular, it identifies when the processes are required and when they're optional.

Send Pending LDAP Requests

You must always run Send Pending LDAP Requests after loading worker records. When you run Send Pending LDAP Requests:

- User accounts are created for new workers.
- Roles are provisioned to user accounts as specified by current role mappings.

If appropriate role mappings don’t exist when you load new workers, then user accounts are created but no roles are provisioned. User accounts without roles are automatically suspended when Send Pending LDAP Requests completes. Also, subsequent role-provisioning requests to the suspended user accounts may be slow to process. To avoid this scenario:

1. Create a role mapping to provision at least one role, for example, Employee, to every current worker before you load worker records.
2. Prevent the creation of user accounts for terminated workers. Set the User Account Creation for Terminated Workers option to No using the following in the Setup and Maintenance work area:
   - Functional Area: Workforce Structures
   - Task: Manage Enterprise HCM Information
3. Load worker records.

Note: Always create a role mapping for the workers you’re loading before you load them. Having this role-provisioning rule in place prevents new user accounts from being suspended unnecessarily.

Autoprovision Roles for All Users

You don’t have to run this process after the initial or only load of worker records if you create role mappings to autoprovision roles before you load workers. However, if you load the workers before creating the role mapping, then you must run this process followed by Send Pending LDAP Requests for the loaded workers.
You don’t have to run this process after an incremental update. However, you’re recommended to run this process whenever a change is made to the autoproposniong rules.

Send Personal Data for Multiple Users to LDAP

Run this process after the initial or only load of worker records. Run it after an incremental load:

- For changed users only
- If you’re loading workers separately from their managers
- If updates were made to personal data by means of SQL scripts

Related Topics

- Role Mappings
- Create a Role Mapping

How You Validate Loaded Data

Once HCM Data Loader loads your data successfully, you’re recommended to validate that the loaded objects appear as expected. This topic identifies some of the ways in which you can validate loaded data. These include use of:

- Manual testing
- Oracle Transactional Business Intelligence (OTBI) reports
- HCM Extracts

Using Manual Testing

You can:

- Validate data interactively by searching for a representative sample. For example, you can search for workers in the Directory or Manage Person work area. If a worker is also a manager, then you can explore the manager hierarchy.
- Validate work structures by searching for them. For example, you can search for jobs on the Manage Job page and locations on the Manage Locations page. Alternatively, you can test that they’re available for selection. For example, you can update a worker’s assignment to select a different job or location.

Using Oracle Transactional Business Intelligence Reports

You can use predefined OTBI reports to validate some uploaded data. To run these reports:

1. Open the Reports and Analytics work area.
2. In the Contents pane of the work area, open the folder Shared Folders > Human Capital Management. Many subfolders of the Human Capital Management folder contain a Transactional Analysis Samples folder, which contains the predefined reports.

To list the predefined reports for an area, select the relevant Transactional Analysis Samples folder in the Contents pane and click the Browse Catalog icon. A list of predefined reports for the selected area, such as Workforce Management, appears in the work area. For example, you can run the Job Details report to list active jobs and the Location Details report to list active locations.
You can also design your own reports using Oracle BI Publisher. To open Oracle BI Publisher, select **Create > Report** in the Contents pane of the Reports and Analytics work area.

### Using HCM Extracts

You can use HCM Extracts to extract data from Oracle HCM Cloud and deliver it in various formats. These formats include CSV, XML, Microsoft Excel, HTML, RTF, and PDF.
Purging Person Data from Test Environments

Enable the Purge Person Data in Test Environments Process

During implementation and testing, you may want to load, test, purge, and reload HCM data. To purge person data from test environments, use the Purge Person Data in Test Environments process. This topic explains how to enable the process and identifies the data that it purges.

Enabling the Purge Process for Person Data

To use the Purge Person Data in Test Environments process, you must raise a service request (SR). In response, Oracle Support supplies a key to enable the process. When you have the key, which is specific to your environment, follow these steps:

1. In the Setup and Maintenance work area, go to the following for your offering:
   - Functional Area: HCM Data Loader
   - Task: Configure HCM Data Loader
2. On the Configure HCM Data Loader page, enter the key from Oracle Support in the Value field for the Purge Person Enabled Key parameter.
3. Click Save and Close.

You can now run the Purge Person Data in Test Environments process.

Note: You can't run the Purge Person Data in Test Environments process in production environments.

Purging Person Data

The Purge Person Data in Test Environments process purges person records, including passports, citizenship records, licenses, emails, addresses, national identifiers, and extra information type data. It also purges dependent data of the following types:

- Absences
- Assignments, including extra information type data
- Benefits
- Calculation cards
- Compensation
- Contacts and contact relationships
- Documents and attachments
- Element entries
- Goals
Deletion requests are generated for user accounts in the test environment. To process these requests, you must run the Send Pending LDAP Requests process when Purge Person Data in Test Environments completes.

Exclusions from the Purge Process

The Purge Person Data in Test Environments process doesn’t purge data of the following types:

- Predefined person data.
- Person data that was loaded using HCM File-Based Loader.
- Oracle Trading Community Architecture (TCA) data.

The process inactivates dependent TCA records in the TCA tables but doesn’t purge them.

If processed payroll actions are associated with a worker, then the worker isn’t purged. You must roll back relevant payroll processes before you can purge such workers.

Purge Person Data from Test Environments

This topic describes how to run the Purge Person Data in Test Environments process. The process must be enabled for use in your test environment. This topic also describes how to run Send Pending LDAP Requests to process user-deletion requests generated by the Purge Person Data in Test Environments process.

Run the Purge Person Data in Test Environments Process

You must have the Purge Person Data in Test Environments function security privilege to run this process. The predefined Human Capital Management Integration Specialist job role has this privilege.

Sign in and follow these steps:

2. In the Scheduled Processes work area, click Schedule New Process.
3. In the Schedule New Process dialog box, search for and select the Purge Person Data in Test Environments process.
4. In the Process Details dialog box, set one of the following parameters to identify the person data to purge.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person ID</td>
<td>Provide one or more person IDs. Separate multiple IDs with commas, for example, 100010034148157, 100010034148169, 100010034148181, 100010034148192.</td>
</tr>
<tr>
<td>Person Number</td>
<td>Provide one or more person numbers. Separate multiple numbers with commas. You can use the wildcard character to supply partial values, for example, PER%, Test%,%abc%.</td>
</tr>
</tbody>
</table>
To return the person IDs of all person records where the last name starts with the characters `John`, you could use the following SQL query:

```
SELECT papf.person_id FROM per_all_people_f papf, per_person_names_f ppnf WHERE ppnf.last_name like 'John%' and ppnf.person_id = papf.person_id
```

To return all person IDs in the test environment, use the following SQL query:

```
SELECT person_id FROM per_all_people_f
```

5. Set the **Save** parameter, as follows.

<table>
<thead>
<tr>
<th>Save Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Lists the persons whose data would be purged by the process. No data is purged.</td>
</tr>
<tr>
<td>Y</td>
<td>Purges the person data.</td>
</tr>
</tbody>
</table>

6. Click **Submit**.

**Tip:** The time this process takes depends on the amount of data to be purged. Don’t cancel the process, as its final stages include clean-up operations. If you cancel the process, then the clean-up operations resume when you next run the process.

---

**Run the Send Pending LDAP Requests Process**

**Purge Person Data in Test Environments** generates requests to delete user accounts from your LDAP directory. To process these requests, run the **Send Pending LDAP Requests** process. If this process isn’t scheduled to run in the test environment, then follow these steps when **Purge Person Data in Test Environments** completes:

1. In the Scheduled Processes work area, click **Schedule New Process**.
2. In the Schedule New Process dialog box, search for and select the **Send Pending LDAP Requests** process.
3. In the Process Details dialog box:
   a. Set **User Type** to **All**.
   b. Leave **Batch Size** set to **A**, which means that the processing batch size is calculated automatically.
4. Click **Submit**.
Tables Purged by the Purge Person Data in Test Environments Process

This topic lists the tables that are purged when you run the **Purge Person Data in Test Environments** process.

<table>
<thead>
<tr>
<th>Product</th>
<th>Table</th>
</tr>
</thead>
</table>
| Absence Management  | • ANC_PER_ABS_CERTS  
                      | • ANC_PER_ABS_ENTRIES  
                      | • ANC_PER_ABS_ENTRY_DTLS  
                      | • ANC_PER_ABS_MATERNITY  
                      | • ANC_PER_ABS_PLAN_ENTRIES  
                      | • ANC_PER_ABS_PLN_SUMM_ENT  
                      | • ANC_PER_ABS_TYPE_ENTRIES  
                      | • ANC_PER_ACCRUAL_ENTRIES  
                      | • ANC_PER_ACRRL_ENTRY_DTLS  
                      | • ANC_PER_EVENTS  
                      | • ANC_PER_PLAN_ENROLLMENT  |
| Benefits            | • BEN_BATCH_ACTN_ITEM_INFO  
                      | • BEN_BATCH_BNFT_CERT_INFO  
                      | • BEN_BATCH_COMMU_INFO  
                      | • BEN_BATCH_DPNT_INFO  
                      | • BEN_BATCH_ELCTBL_CHC_INFO  
                      | • BEN_BATCH_ELIG_INFO  
                      | • BEN_BATCH_LER_INFO  
                      | • BEN_BATCH_RATE_INFO  
                      | • BEN_BENEFIT_ACTIONS  
                      | • BEN_BENEFIT_RELATIONS_F  
                      | • BEN_CVRD_DPNT_CTFN_PRVDD  
                      | • BEN_ELCTBL_CHC_CTFN  
                      | • BEN_ELIG_CVRD_DPNT  
                      | • BEN_ELIG_DPNT  
                      | • BEN_ELIG_FLX_CRDT_POOL_CHC  
                      | • BEN_ELIG_PER_ELCTBL_CHC  
                      | • BEN_ELIG_PER_F  
                      | • BEN_ELIG_PER_OPT_F  
                      | • BEN_ELIG_RSLT_F  
                      | • BEN_ENRT_BNFT  
                      | • BEN_ENRT_RT  
                      | • BEN_EXTRACTDETAILS_ARCH  
                      | • BEN_EXTRACT_REQ_DETAILS  
                      | • BEN_FLEX_CRDT_LDGR  
                      | • BEN_PER_BENEFIT_GROUP_F  
                      | • BEN_PER_BINF_ORG  
                      | • BEN_PER_BNFTS_BAL_F  
                      | • BEN_PER_IN_LER  
                      | • BEN_PER_LE_HABITS_COV_F  
                      | • BEN_PERSON_ACTIONS |
## Purging Person Data from Test Environments

<table>
<thead>
<tr>
<th>Product</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Human Capital Management Cloud</td>
<td></td>
</tr>
<tr>
<td>Integrating with HCM</td>
<td>Chapter 29</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Human Capital Management Cloud</td>
<td></td>
</tr>
<tr>
<td>Integrating with HCM</td>
<td>Chapter 29</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compensation Management</td>
<td></td>
</tr>
<tr>
<td>• BEN_PL_ELCTBL_CHC_POPL</td>
<td></td>
</tr>
<tr>
<td>• BEN_PL_BNF</td>
<td></td>
</tr>
<tr>
<td>• BEN_PL_BNF_CTFN_PRVDD</td>
<td></td>
</tr>
<tr>
<td>• BEN_PRRTT_ENRT_ACTN</td>
<td></td>
</tr>
<tr>
<td>• BEN_PRRTT_ENRT_CTFN_PRVDD</td>
<td></td>
</tr>
<tr>
<td>• BEN_PRRTT_ENRT_RSLT</td>
<td></td>
</tr>
<tr>
<td>• BEN_PRRTT_LEG_DISCMR_ACTN</td>
<td></td>
</tr>
<tr>
<td>• BEN_PRRTT_RT_VAL</td>
<td></td>
</tr>
<tr>
<td>• BEN_PTNL_LER_FOR_PER</td>
<td></td>
</tr>
<tr>
<td>• BEN_REL_ASSIGNMENTS_F</td>
<td></td>
</tr>
<tr>
<td>• BEN_REPORTING</td>
<td></td>
</tr>
</tbody>
</table>

| Compensation Management |       |
| • CMP_BUDGET_AUDIT |       |
| • CMP_BUDGET_USAGES |       |
| • CMP_CWB_ALERTS |       |
| • CMP_CWB_APPROVAL_HISTORY |       |
| • CMP_CWB_AUDIT |       |
| • CMP_CWB_FEEDBACK |       |
| • CMP_CWB_HRCHY |       |
| • CMP_CWB_INTCPT_APPR_DTLS |       |
| • CMP_CWB_MODEL_B |       |
| • CMP_CWB_MODEL_DTL |       |
| • CMP_CWB_MODEL_TL |       |
| • CMP_CWB_PERF_RATINGS |       |
| • CMP_CWB_PERSON_DEFAULTS |       |
| • CMP_CWB_PERSON_ELEMENTS |       |
| • CMP_CWB_PERSON_INFO |       |
| • CMP_CWB_PERSON_RATES |       |
| • CMP_CWB_POST_LOG |       |
| • CMP_CWB_POST_PERSON |       |
| • CMP_CWB_PROMOTIONS |       |
| • CMP_CWB_RANKS |       |
| • CMP_CWB_SUMMARY |       |
| • CMP_CWB_SWITCH_MANAGERS |       |
| • CMP_DYNAMIC_FILTER |       |
| • CMP_INTERCEPT_APPROVERS |       |
| • CMP_PERSON_BUDGETS |       |
| • CMP_PROCESS_DATA |       |
| • CMP_PROCESS_RUN_INFO |       |
| • CMP_SALARY |       |
| • CMP_SALARY_COMPONENTS |       |
| • CMP_SALARY_EXPORT |       |
| • CMP_STOCK_DETAILS |       |
| • CMP_TCS_PER_ITEM_VALUE |       |
| • CMP_TCS_PER_PERD |       |
| • CMP_TCS_PER_PERD_STMT |       |
| • CMP_TCS_PER_PERD_STMT_CAT |       |
| • CMP_TCS_PER_ROW_VALUES |       |
| • CMP_TCS_REPORT_DETAILS |       |
| • CMP_VC_PER_ELIG_PLANS |       |
| • CMP_VC.PER_ELIG_PLANS |       |

| Oracle Middleware Extensions for Applications |       |
| • FND_ATTACHEDDOCUMENTS |       |
| • FND_DOCUMENTS |       |
### Product Table

<table>
<thead>
<tr>
<th>Product</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• FND_DOCUMENTS_TL</td>
</tr>
<tr>
<td></td>
<td>• FND_SESSION_USERS</td>
</tr>
<tr>
<td>Workforce Modeling</td>
<td>• HMO_MODEL_PLANDETAILS</td>
</tr>
<tr>
<td></td>
<td>• HMO_MODEL_PLANS_B</td>
</tr>
<tr>
<td></td>
<td>• HMO_MODEL_PLANS_TL</td>
</tr>
<tr>
<td>Performance Management</td>
<td>• HRA_EVAL_ITEMS</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVAL_PARTICIPANTS</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVAL_RATINGS</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVAL_ROLE_ACTIONS</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVAL_ROLES</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVAL_SECTIONS</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVAL_STEPS</td>
</tr>
<tr>
<td></td>
<td>• HRA_EVALUATIONS</td>
</tr>
<tr>
<td>Goal Management</td>
<td>• HRG_GOAL_ACCESS</td>
</tr>
<tr>
<td></td>
<td>• HRG.GOAL_ACTIONS</td>
</tr>
<tr>
<td></td>
<td>• HRG.GOAL ALIGNMENTS</td>
</tr>
<tr>
<td></td>
<td>• HRG.GOAL_TARGET_OUTCOMES</td>
</tr>
<tr>
<td></td>
<td>• HRG.GOALS</td>
</tr>
<tr>
<td>Succession Management</td>
<td>• HRM_PLANS</td>
</tr>
<tr>
<td></td>
<td>• HRM_PLAN_CANDIDATES</td>
</tr>
<tr>
<td></td>
<td>• HRM_PLAN_OWNERS</td>
</tr>
<tr>
<td>Talent Review</td>
<td>• HRR_DASHBOARDS</td>
</tr>
<tr>
<td></td>
<td>• HRR_MEETINGS</td>
</tr>
<tr>
<td></td>
<td>• HRR_MEETING_FACILITATORS</td>
</tr>
<tr>
<td></td>
<td>• HRR_MEETING_PARTICIPANTS</td>
</tr>
<tr>
<td></td>
<td>• HRR_MEETING_REVIEWEES</td>
</tr>
<tr>
<td>Questionnaire</td>
<td>• HRQ_QSTN_RESPONSES</td>
</tr>
<tr>
<td></td>
<td>• HRQ_QSTNR_PARTICIPANTS</td>
</tr>
<tr>
<td></td>
<td>• HRQ_QSTNR_RESPONSES</td>
</tr>
<tr>
<td>Profile Management</td>
<td>• HRT_NOTES</td>
</tr>
<tr>
<td></td>
<td>• HRT_POOL_MEMBERS</td>
</tr>
<tr>
<td></td>
<td>• HRT_POOL_OWNERS</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILE_INTERESTS</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILE_ITEMS</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILE_KEYWORDS</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILE_MATCH_CRIT</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILE_RELATIONS</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILES_B</td>
</tr>
<tr>
<td></td>
<td>• HRT_PROFILES_TL</td>
</tr>
<tr>
<td>Workforce Management</td>
<td>• HWM_TM_REC</td>
</tr>
<tr>
<td></td>
<td>• HWM_TM_RECPH</td>
</tr>
<tr>
<td></td>
<td>• HWM_TM_RECPH_GRP</td>
</tr>
<tr>
<td></td>
<td>• HWM_TM_RECPH_GRP_SUM</td>
</tr>
<tr>
<td></td>
<td>• HWM_TM_RECPH_GRP_USAGES</td>
</tr>
<tr>
<td></td>
<td>• HWM_TM_RECPH_GRP_USAGES</td>
</tr>
<tr>
<td>Workforce Predictions</td>
<td>• HWP_ARCH_DATA_MINING_REASONS</td>
</tr>
<tr>
<td></td>
<td>• HWP_ARCH_DATA_MINING_RESULTS</td>
</tr>
</tbody>
</table>
### Global Payroll

- PAY_ASSIGNED_PAYROLL
- PAY_ASSIGNED_PAYROLLS_DN
- PAY_ASSIGNED_PAYROLLS_DN_
- PAY_ASSIGNED_PAYROLLS_F
- PAY_ASSIGNED_PAYROLLS_F_
- PAY_DATES
- PAY_DIR_CARD_COMPONENTS_F
- PAY_DIR_CARD_COMPONENTS_F_
- PAY_DIR_CARDS_F
- PAY_DIR_CARDS_F_
- PAY_DIR_COMP_DETAILS_F
- PAY_DIR_COMP_DETAILS_F_
- PAY_DIR_REP_CARD_USAGES_F
- PAY_DIR_REP_CARD_USAGES_F_
- PAY_DIR_REP_CARDS_F
- PAY_DIR_REP_CARDS_F_
- PAY_ELEMENT_ENTRIES_F
- PAY_ELEMENT_ENTRIES_F_
- PAY_ELEMENT_ENTRY_VALUES_F
- PAY_ELEMENT_ENTRY_VALUES_F_
- PAY_ENTRY_PCT_DIST
- PAY_ENTRY_PROC_DETAILS
- PAY_ENTRY_USAGES
- PAY_PAY_RELATIONSHIPS_DN
- PAY_PAY_RELATIONSHIPS_F
- PAY_PERSON_PAY_METHODS_F_
- PAY_RANGE_ITEMS_F
- PAY_RANGE_ITEMS_F_
- PAY_REL_GROUPS_DN
- PAY_REL_GROUPS_F
- PAY_RETRO_ENTRIES
- PAY_VALUE_DEFINITIONS_F
- PAY_VALUE_DEFINITIONS_F_

### Global Human Resources

- HR_DOCUMENT_DELIVERY_PREFS
- HR_DOCUMENTS_OF_RECORD
- PER_ABSENCE_CASES
- PER_ACTION_OCCURRENCES
- PER_ADDRESSES_F
- PER_ALL_ASSIGNMENTS_M
- PER_ALL_PEOPLE_F
- PER_ALLOCATED_CHECKLISTS
- PER_ALLOCATED_TASKS
- PER_ASG_RESPONSIBILITIES
- PER_ASSIGN_GRADE_STEPS_F
- PER_ASSIGN_WORK_MEASURES_F
- PER_ASSIGNMENT_EXTRA_INFO_M
- PER_ASSIGNMENT_SUPERVISORS_F
### Purging Person Data from Test Environments

<table>
<thead>
<tr>
<th>Product</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PER_CITIZENSHIPS</td>
</tr>
<tr>
<td></td>
<td>PER_CONTACT_EXTRA_INFO_F</td>
</tr>
<tr>
<td></td>
<td>PER_CONTACT_RELSHIPS_F</td>
</tr>
<tr>
<td></td>
<td>PER_CONTRACTS_F</td>
</tr>
<tr>
<td></td>
<td>PER_DELEGATED_ROLES</td>
</tr>
<tr>
<td></td>
<td>PER_DISABILITIES_F</td>
</tr>
<tr>
<td></td>
<td>PER_DRIVERS_LICENSE_TYPES</td>
</tr>
<tr>
<td></td>
<td>PER_DRIVERS_LICENSES</td>
</tr>
<tr>
<td></td>
<td>PER_EMAIL_ADDRESSES</td>
</tr>
<tr>
<td></td>
<td>PER_ETHNICITIES</td>
</tr>
<tr>
<td></td>
<td>PER_IMAGES</td>
</tr>
<tr>
<td></td>
<td>PER_KEYWORDS</td>
</tr>
<tr>
<td></td>
<td>PER_LDAP_REQUESTS</td>
</tr>
<tr>
<td></td>
<td>PER_LDAP_ROLE_MEMBERSHIPS</td>
</tr>
<tr>
<td></td>
<td>PER_LDAP_USERS</td>
</tr>
<tr>
<td></td>
<td>PER_MANAGER_HRCHY_DN</td>
</tr>
<tr>
<td></td>
<td>PER_NATIONAL_IDENTIFIER</td>
</tr>
<tr>
<td></td>
<td>PER_PASSPORTS</td>
</tr>
<tr>
<td></td>
<td>PER_PEOPLE_EXTRA_INFO_F</td>
</tr>
<tr>
<td></td>
<td>PER_PEOPLE_LEGISLATIVE_F</td>
</tr>
<tr>
<td></td>
<td>PER_PERIODS_OF_SERVICE</td>
</tr>
<tr>
<td></td>
<td>PER_PERSON_ADDR_USAGES_F</td>
</tr>
<tr>
<td></td>
<td>PER_PERSON_DLVRY_METHODS</td>
</tr>
<tr>
<td></td>
<td>PER_PERSON_NAMES_F</td>
</tr>
<tr>
<td></td>
<td>PER_PERSON_TYPE_USAGES_M</td>
</tr>
<tr>
<td></td>
<td>PER_PEOPLE</td>
</tr>
<tr>
<td></td>
<td>PER_PHONES</td>
</tr>
<tr>
<td></td>
<td>PER_PORTRAIT_CARD_CONFIG</td>
</tr>
<tr>
<td></td>
<td>PER_PORTRAIT_PREFERENCES</td>
</tr>
<tr>
<td></td>
<td>PER_RELIGIONS</td>
</tr>
<tr>
<td></td>
<td>PER_SCHEDULE_ASSIGNMENTS</td>
</tr>
<tr>
<td></td>
<td>PER_SHARE_INFORMATION</td>
</tr>
<tr>
<td></td>
<td>PER_USER_HISTORY</td>
</tr>
<tr>
<td></td>
<td>PER_USER_ROLES</td>
</tr>
<tr>
<td></td>
<td>PER_USERS</td>
</tr>
<tr>
<td></td>
<td>PER_VISAS_PERMITS_F</td>
</tr>
</tbody>
</table>
HCM Extracts and Diagnostics for HCM Data Loader and HCM Spreadsheet Data Loader

HCM Data Loader Extracts

HCM Extracts provides four extracts for HCM Data Loader, as shown in this table.

<table>
<thead>
<tr>
<th>Extract Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCM Data Loader Data Set Summary</td>
<td>Provides status values and record counts for both a specified data set and each business object in the data set. It reports all error messages and identifies all records affected by each error.</td>
</tr>
<tr>
<td>Compensation Changes Extract</td>
<td>For a specified compensation run, extracts assignment, salary, and bonus changes made by the run. This extract is most useful if you’re implementing a coexistence scenario.</td>
</tr>
<tr>
<td>Integration Object User Key Refresh</td>
<td>Retrieves latest user key values. Run this extract before running the Integration Object User Key Map Extract.</td>
</tr>
<tr>
<td>Integration Object User Key Map Extract</td>
<td>Extracts GUIDs, source keys, surrogate IDs, and user keys for a specified business object hierarchy.</td>
</tr>
</tbody>
</table>

The output from all extracts is in **XML** format by default. You can provide a BI Publisher layout to reformat the output. You can also define options to automate delivery of the output.

Before users can run any of these extracts, you:

1. Copy the predefined extract, specify a delivery option, and compile the formulas for your copy of the extract.
2. Configure your copy of the extract.

Once the extract is configured, users can run it and view the output.

Setting Up HCM Data Loader Extracts

Manage an HCM Data Loader Extract Definition

HCM Extracts provides four predefined extracts for HCM Data Loader:

- HCM Data Loader Data Set Summary
- Compensation Changes Extract
- Integration Object User Key Refresh
- Integration Object User Key Map Extract
This topic explains how to copy a predefined extract, specify a delivery option, and compile the formulas for the copied extract. You can follow these instructions for any of the HCM Data Loader extracts.

Copy the Extract

Follow these steps to copy an HCM Data Loader extract:

1. On the Tasks panel tab of the Data Exchange work area, select Manage Extract Definitions.
2. On the Manage HCM Extract Definitions page, search for the extract.
3. Select the extract in the search results and click Copy.
4. In the Copy Extract Definition dialog box, enter a name for your copy of the extract and click OK.
   A confirmation message appears when the copy is complete.

Specify a Delivery Option for the Copied Extract

Follow these steps to specify a delivery option:

1. On the Manage HCM Extract Definitions page, search for your copy of the extract.
2. Click the extract name in the search results to display the extract definition on the Manage HCM Extract Definitions page.
3. Click Deliver.
4. In the Extract Delivery Options section, click Add.
5. Specify a delivery option. To keep the default XML output, set the Output Type to Data.
6. For the XML delivery option, specify Delivery Option Name, Output Name, and Delivery Type values.

   Tip: The values that you must specify vary by output type.

7. Click Save.

For more information, see Oracle Fusion HCM Extracts: Delivery Options (document ID 1573099.1) on My Oracle Support at https://support.oracle.com.

Compile the Formulas for the Copied Extract

Follow these steps to compile the formulas:

1. On the Manage HCM Extract Definitions page, click Validate.
2. In the Extract Execution Tree section, click Compile All Formula to compile the formulas.
3. Refresh the table periodically until all formulas are successfully compiled. A check mark means success.
4. Click Submit.

Once these steps are complete, perform the Refine HCM Extracts task in the Data Exchange work area to configure the copied HCM Data Loader extract.

Related Topics
- Oracle Fusion HCM Extracts: Delivery Options

Configure a Copied HCM Data Loader Extract

HCM Extracts provides four predefined extracts for HCM Data Loader:

- HCM Data Loader Data Set Summary
This topic explains how to configure an extract after copying it, specifying a delivery option, and compiling its formulas. Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select **Refine Extracts** to open the Manage Payroll Flow Patterns page.
2. On the Manage Payroll Flow Patterns page, enter the name of your copy of the extract in the **Flow Pattern** field and click **Search**.
3. In the search results, select your extract and click **Edit**.
4. Click the Parameters tab to review the parameter definitions.
5. Edit the **Display**, **Sequence**, **Parameter Basis**, and **Basis Value** for each parameter. You can also edit the parameter names, if required.
6. Save your changes.

Users can now run the extract.

### HCM Data Loader Data Set Summary Extract Parameters

This topic describes how to configure the parameters of the HCM Data Loader Data Set Summary extract. To configure the parameters, you perform the **Refine Extracts** task in the Data Exchange work area.

#### Configuring the Parameters

This table shows the parameters that you must edit and the required values for the HCM Data Loader Data Set Summary extract. You can ignore parameters not shown here.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Display</th>
<th>Sequence</th>
<th>Parameter Basis</th>
<th>Basis Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective Date</td>
<td>No</td>
<td>3</td>
<td>Context Binding</td>
<td>System Date</td>
</tr>
<tr>
<td>Start Date</td>
<td>No</td>
<td>9</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Content ID</td>
<td>Yes</td>
<td>12</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Data Set Name</td>
<td>Yes</td>
<td>13</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Process ID</td>
<td>Yes</td>
<td>11</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Show Messages</td>
<td>No</td>
<td>14</td>
<td>Constant Bind</td>
<td>Y</td>
</tr>
<tr>
<td>Show Stack Trace</td>
<td>No</td>
<td>15</td>
<td>Constant Bind</td>
<td>N</td>
</tr>
</tbody>
</table>
Compensation Changes Extract Parameters

This topic describes how to configure the parameters of the Compensation Changes Extract for HCM Data Loader.

To configure the parameters, you perform the **Refine Extracts** task in the Data Exchange work area.

### Configuring the Parameters

This table shows the parameters that you must edit and the required values for the Compensation Changes Extract. You can ignore parameters not shown here.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Display</th>
<th>Sequence</th>
<th>Parameter Basis</th>
<th>Basis Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective Date</td>
<td>No</td>
<td>3</td>
<td>Context Binding</td>
<td>System Date</td>
</tr>
<tr>
<td>Start Date</td>
<td>No</td>
<td>9</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Extract Target</td>
<td>No</td>
<td>15</td>
<td>Constant Bind</td>
<td>HDL</td>
</tr>
<tr>
<td>Period Name</td>
<td>Yes</td>
<td>12</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Plan</td>
<td>Yes</td>
<td>11</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Process Id</td>
<td>Yes</td>
<td>13</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Run Id</td>
<td>Yes</td>
<td>14</td>
<td>Not applicable</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

Integration Object User Key Refresh and Integration Object User Key Map Extract Parameters

This topic describes how to configure the parameters of these HCM Data Loader extracts:

- Integration Object User Key Refresh
- Integration Object User Key Map Extract

To configure the parameters, you perform the **Refine Extracts** task in the Data Exchange work area.

### Configuring the Parameters

This table shows the parameters that you must edit and the required values for the Integration Object User Key Refresh extract.
### Parameter | Display | Sequence | Parameter Basis | Basis Value
---|---|---|---|---
Effective Date | Mandatory | 3 | Context Binding | System Date
Start Date | No | 9 | Not applicable | Not applicable
Process Configuration Group | Yes | 17 | Not applicable | Not applicable
Business Object | Yes | 76 | Not applicable | Not applicable
Component Discriminator | Yes | 81 | Not applicable | Not applicable
Include Child Components | Yes | 91 | Not applicable | Not applicable

*Note:* You’re recommended to change the name of the **Business Object** parameter to **Business Object Discriminator**.

This table shows the parameters that you must edit and the required values for the Integration Object User Key Map Extract.
Note: You're recommended to change the name of the Business Object parameter to Business Object Discriminator.

Running HCM Data Loader Extracts

Submit an HCM Data Loader Extract

HCM Extracts provides four predefined extracts for HCM Data Loader:
- HCM Data Loader Data Set Summary
- Compensation Changes Extract
- Integration Object User Key Refresh
- Integration Object User Key Map Extract

This topic explains how to submit a configured HCM Data Loader extract.

Select the Extract

Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select Submit Extracts to open the Submit a Process or Report page.
2. Using the query by example fields, find your extract and click Next.
   The Enter Parameters page opens.
3. In the Payroll Flow field, enter a flow name for this run of the extract.
4. Enter the parameter values for your extract.

Enter Parameters for the HCM Data Loader Data Set Summary Extract

This table shows the parameters for the HCM Data Loader Data Set Summary extract. Enter any one of the Content ID, Process ID, and Data Set Name parameters. You can find all of these values for a specific data set on the Import and Load Data page.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content ID</td>
<td>The content ID associated with the data set. This ID was assigned when you uploaded your source file to the Oracle WebCenter Content server.</td>
</tr>
<tr>
<td>Data Set Name</td>
<td>The data set name, which is also your source file name.</td>
</tr>
<tr>
<td>Process Configuration Group</td>
<td>Leave blank.</td>
</tr>
<tr>
<td>Process ID</td>
<td>The process ID of the process that created the data set. This ID was assigned when you submitted your source file for processing by HCM Data Loader.</td>
</tr>
</tbody>
</table>
Enter Parameters for the Compensation Changes Extract

This table shows the parameters of the Compensation Changes Extract. To identify the compensation run to report on, use one of the following parameters or pairs of parameters:

- Plan and Period Name
- Process Id
- Run Id

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan</td>
<td>The compensation plan name</td>
</tr>
<tr>
<td>Period Name</td>
<td>The period name for the compensation run</td>
</tr>
<tr>
<td>Process Id</td>
<td>The process ID for the compensation run</td>
</tr>
<tr>
<td>Run Id</td>
<td>The run ID for the compensation run</td>
</tr>
</tbody>
</table>

Enter Parameters for the Integration Object User Key Refresh Extract

This table shows the parameters of the Integration Object User Key Refresh Extract.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Object</td>
<td>The file discriminator of the business object for which you want to refresh keys. For example, for some or all components of the Worker object, set this parameter to Worker. You can find the file discriminator values on the Business Object Details page for the object in the Data Exchange work area.</td>
</tr>
<tr>
<td>Component Discriminator</td>
<td>The file discriminator of the business object component. For example, to refresh keys for the Work Relationship component of the Worker object, set this parameter to WorkRelationship. You can find the file discriminator values on the Business Object Details page for the object in the Data Exchange work area.</td>
</tr>
<tr>
<td>Include Child Components</td>
<td>Set to Yes to include the child components of the business object or component in the refresh.</td>
</tr>
<tr>
<td>Process Configuration Group</td>
<td>Leave blank.</td>
</tr>
</tbody>
</table>

Enter Parameters for the Integration Object User Key Map Extract

This table shows the parameters of the Integration Object User Key Map Extract.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Object</td>
<td>The file discriminator of the business object for which you want to extract keys. For example, for some or all components of the Worker object, set this parameter to Worker. You can find the file discriminator values on the Business Object Details page for the object in the Data Exchange work area.</td>
</tr>
</tbody>
</table>
Parameter | Description
--- | ---
Component Discriminator | The file discriminator of the business object component. For example, to extract keys for the Work Relationship component of the Worker object, set this parameter to WorkRelationship. You can find the file discriminator values on the Business Object Details page for the object in the Data Exchange work area.

Include Child Components | Set to Yes to include the child components of the business object or component in the extract.

Tip: If you’re extracting a large number of records, then you’re recommended to set this parameter to No and extract each component separately.

Process Configuration Group | Leave blank.

Source System Owner | If required, select a value to extract the keys for a specified source system owner. The extract includes all types of keys, but only for those object instances with this source system owner.

Submit the Extract
Follow these steps:

1. Click **Next** until you reach the Review page.
2. Click **Submit**.

A confirmation message appears when your extract is submitted successfully.

3. Close the Confirmation dialog box.

You can now view the extract output.

View the Results of an HCM Data Loader Extract

HCM Extracts provides four predefined extracts for HCM Data Loader:

- HCM Data Loader Data Set Summary
- Compensation Changes Extract
- Integration Object User Key Refresh
- Integration Object User Key Map Extract

This topic explains how to view the output from an HCM Data Loader extract. Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select **View Extract Results** to open the View Extract Run Details page.
2. Search for the extract.
3. In the search results, click the name of the extract to open the View Extract Results tab.
4. In the Extract Runs section, use the query by example fields to find the extract run for which you want the results.
5. In the Details section, click the Extract Delivery Options tab. This tab appears only if a delivery option was specified when the extract was configured.
6. Click **Download** to download the extract results.
Diagnostic Tests for HCM Data Loader

HCM Data Loader Error Analysis Report

The HCM Data Loader Error Analysis Report is a diagnostic test. It provides technical information that can help to resolve data-loading errors. Run this diagnostic test when error messages don’t supply enough information for you to identify a solution. You can send the test output to Oracle Support, if necessary.

To run the HCM Data Loader Error Analysis Report test, select Settings and Actions > Run Diagnostic Tests.

Diagnostic Test Parameters

Content ID

A unique reference to a single data set. You can find the content ID in the Content ID column of the Search Results table on the Import and Load Data page. You must specify either the content ID or the process ID to identify the data set.

Process ID

A unique reference to a single data set. You can find the process ID in the Process ID column of the Search Results table on the Import and Load Data page. You must specify either the content ID or the process ID to identify the data set.

Business Object Name

The name of a business object in the data set, such as Worker or Job. You can find the business object name in the Business Object column of the Details table on the Import and Load Data page. This parameter is optional, but you’re recommended to supply it if your data set contains multiple business objects.

Detailed Mode

Set this parameter to Y to include the details of any reported errors in the output, including message stack trace for the records in error.

Diagnostic Test Results

The diagnostic test output is in user-readable HTML format. It provides:

- Status information
- Details of every process submitted to process your data set
- All errors raised for each business object in the data set
- Message stack trace and nonsensitive attribute values for the records in error, if you set the Detailed Mode parameter to Y

No sensitive data appears in the test results.
Run the HCM Data Loader Error Analysis Report

The HCM Data Loader Error Analysis Report is a diagnostic test that provides technical information about data loads to help you resolve errors. To submit this diagnostic test, you must have access to the Diagnostic Dashboard.

Follow these steps:

1. Select Settings and Actions > Run Diagnostic Tests to open the Diagnostic Dashboard.
2. In the Search for Tests section of the Diagnostic Dashboard, enter HCM Data Loader Error Analysis Report in the Test Name field and click Search.
3. In the search results, select the check box next to the test name and click Add to Run. The test appears in the Choose Tests to Run and Supply Inputs section of the page.
4. To review the parameters, click the Click to Supply or Edit Input Parameters icon in the Status column.
5. In the Input Parameters dialog box, select parameters to include in this test run and supply values for the selected parameters.

\[\text{\textbf{Note:}}\] Use the parameters to restrict the output from the test to those errors that need investigation. For example, don’t run the test for an entire data set if the errors relate to a single business object.

6. Click OK.
7. If required, enter a name for the test run in the Run Name field. The name can help you to locate your results easily. If you leave the field blank, then a name is generated.
8. Click Run to run the test and click OK to close the confirmation dialog box.
9. In the Diagnostic Test Run Status section, click the Display Latest Test Run Status Information icon to see the results of this test.
10. Expand the test hierarchy for your run to see the results. Click the Report icon to open the report.

You can save the output and attach it to a service request if required.

Diagnostic Tests for HCM Spreadsheet Data Loader

HCM Spreadsheet Data Loader Diagnostic Report

The HCM Spreadsheet Data Loader Diagnostic Report provides a detailed analysis of a spreadsheet template or associated data sets to help you resolve template errors. Run this report when error messages don’t supply enough information for you to resolve any errors. You can send the report output to Oracle Support, if necessary.

To run the HCM Spreadsheet Data Loader Diagnostic Report, select Settings and Actions > Run Diagnostic Tests.

Diagnostic Test Parameters

Data Set Name

To report on a specific data set, enter its name. You must provide a value for either the Data Set Name or the Template Name parameter.

Template Name
To report on a spreadsheet template, enter its name. You must provide a value for either the **Data Set Name** or the **Template Name** parameter.

**Number of Data Sets to Include**

Enter the number of data sets most recently created in spreadsheets generated from the spreadsheet template to include in the report. The default value is 5. This parameter applies only when you also specify a value for the **Template Name** parameter.

**Include Error Details**

Set this parameter to **Y** to provide details, including stack trace, of the most frequent import and load errors in the report output.

**Diagnostic Report Contents**

The report is in user-readable HTML format. It includes the following sections:

<table>
<thead>
<tr>
<th>Report Section</th>
<th>Section Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
<td>Lists all report parameters with the values that you supplied</td>
</tr>
<tr>
<td>Template Details</td>
<td>The values held for the template definition</td>
</tr>
<tr>
<td>Template Business Objects</td>
<td>Details of the business objects that the template supports</td>
</tr>
<tr>
<td>Template Attribute Details</td>
<td>Details of the attributes selected in the template, regardless of whether they’re visible in generated spreadsheets</td>
</tr>
<tr>
<td>Map Table</td>
<td>Details of the template map</td>
</tr>
<tr>
<td>Attribute Map Table</td>
<td>The business object map information</td>
</tr>
<tr>
<td>Flexfield Contexts</td>
<td>Flexfield contexts in the spreadsheet template</td>
</tr>
<tr>
<td>Flexfield Attributes</td>
<td>Flexfield attributes in the spreadsheet template</td>
</tr>
<tr>
<td>Data set Name: &lt;data set name&gt;</td>
<td>One section for each data set in the report</td>
</tr>
</tbody>
</table>

The **Data set Name: <data set name>** section includes the following information for each data set in the report:

- A summary of all errors raised for the data set
- The status and record counts
- Details of the **Import and Load** process that created the data set
- All parameters to the **Import and Load** process and its child processes
Detailed information about each business object in the spreadsheet, including:

- Object status
- Details of the import and load processes
- SET commands
- The 25 most frequent import errors
- The 25 most frequent load errors
- Object and component counts

Run the HCM Spreadsheet Data Loader Diagnostic Report

The HCM Spreadsheet Data Loader Diagnostic Report is a diagnostic test that enables you to analyze problems in custom spreadsheet templates. To run this report, you must have access to the Diagnostic Dashboard.

Follow these steps:

1. Select **Settings and Actions > Run Diagnostic Tests** to open the Diagnostic Dashboard.
2. In the Search for Tests section of the Diagnostic Dashboard, enter **HCM Spreadsheet Data Loader Diagnostic Report** in the **Test Name** field and click **Search**.
3. In the search results, select the check box next to the test name and click **Add to Run**. The test appears in the Choose Tests to Run and Supply Inputs section of the page.
4. To review the parameters, click the **Click to Supply or Edit Input Parameters** icon in the **Status** column.
5. In the Input Parameters dialog box, select parameters to include in this test run and supply values for the selected parameters.

   ✍️ **Note:** Use the parameters to restrict the output from the test to those errors that need investigation. For example, don’t run the test for multiple data sets if the errors relate to a single data set.

6. Click **OK**.
7. If required, enter a name for the test run in the **Run Name** field. The name can help you to locate your results easily. If you leave the field blank, then a name is generated.
8. Click **Run** to run the test and click **OK** to close the confirmation dialog box.
9. In the Diagnostic Test Run Status section, click the **Display Latest Test Run Status Information** icon to see the results of this test.
10. Expand the test hierarchy for your run to see the results. Click the **Report** icon to open the report.

You can save the output and attach it to a service request if required.
31 Best Practices for HCM Data Loader and HCM Spreadsheet Data Loader

HCM Data Loader Best Practices

This topic highlights some best practices for data loading. Follow these recommendations to ensure successful creation and maintenance of data using HCM Data Loader.

Source Keys

The source key is the recommended key type to use for all implementations. If you supply a source key when you create your data, then you can also update or simply reference that data using source keys.

You can’t update some objects if you identify them using only user keys. For example, you can’t update most components of the Worker object if you supply only a user key. The attribute that you want to change is often the attribute that’s used to identify the record. For example, in the Person Address component, the **AddressLine1** attribute is used both to identify the address to update and to supply the new value. Therefore, you’re recommended to use source keys whenever possible.

Data Migration

When creating business objects using HCM Data Loader:

- Deliver one object type in each .zip file. For example, create one .zip file for **jobs**, one for **grades**, one for workers, and so on. To avoid data-reference errors, you must ensure that you load .zip files in the correct order and fix any errors before loading the next file.
- Purge the **stage table** data regularly for **data sets** that you no longer need, especially if you’re loading large volumes of data.
- Understand and follow the rules for creating date-effective data.

**Tip:** HCM Data Loader can load very large volumes of data from a single file. However, do consider the practical challenges, such as error management and the length of the transition period, associated with large data loads. Best practice is to split large loads into logical groupings so that you can extract, transform, and load the data more easily. For example, when loading workers you could load ex-employees separately from current, active employees. If this grouping still leaves you with very large data loads, then you could define subgroups of records based on other significant criteria.

Updating Objects

When updating business objects using HCM Data Loader:

- Understand and follow the rules for maintaining date-effective data. In particular, always include the `SET PURGE_FUTURE_CHANGES N` instruction in the .dat file to ensure that any existing future-dated changes are retained.
- Supply all business-object files in the same .zip file. HCM Data Loader processes them in the correct order. Referenced data is loaded before the data that references it.
Note: If a dependency exists between business-object files, then include the surrogate ID attribute that references the dependent object in the METADATA line. For example, you may be loading both Worker.dat and Job.dat files together. If the Worker object includes references to a new job, then you must include the JobId attribute in the METADATA line of the Worker.dat file. This rule applies even if you’re also supplying user keys. You don’t have to supply a value for the JobId attribute. Having it in the METADATA line enables HCM Data Loader to identify the dependency.

- Supply any changed attribute values and a unique reference to the record that you’re updating. Also, if a dependency exists between the objects in the file, then include the surrogate ID attribute in the METADATA line. Otherwise, don’t include unchanged, optional attributes in the METADATA line of your .dat file. HCM Data Loader validates every attribute name in the METADATA line. Including attributes for which you aren’t providing data has a negative effect on performance.
- Never include both DELETE and MERGE instructions for the same record in the same file. HCM Data Loader doesn’t guarantee the order in which the file lines are processed.

Related Topics
- Source Keys
- Overview of Loading Date-Effective Objects

Overview of Maintaining the Stage Tables

When you load large volumes of data into Oracle HCM Cloud, the stage tables can grow rapidly. You’re recommended to create a schedule to delete processed data sets that you no longer need in the stage tables. How often you delete processed data sets from the stage tables depends on the volume and frequency of your data loads. During data migration, consider deleting every large data set as its processing completes. This topic describes the role of stage tables in data loading and explains how the deletion process maintains the stage tables automatically.
How Data Is Loaded to Oracle HCM Cloud

This figure shows how data is loaded to the application tables. Data that you load using either HCM Data Loader or HCM Spreadsheet Data Loader is imported first to stage tables, where some validation occurs. HCM Spreadsheet Data Loader has its own stage tables, from where valid data is loaded to the HCM Data Loader stage tables. Valid data is loaded from the HCM Data Loader stage tables to the application tables.

How Processed Data Sets Are Deleted from Stage Tables

You can delete processed data sets from both sets of stage tables.

- To delete data sets from the HCM Spreadsheet Data Loader stage tables, use the Delete Spreadsheet Stage Table Data task. It runs the Delete HCM Spreadsheet Data Loader Stage Table Data process.
- To delete data sets from the HCM Data Loader stage tables, use the Delete Stage Table Data task. It runs the Delete HCM Data Loader Stage Table Data process.

Perform both tasks in the Data Exchange work area.

You can delete:

- Individual data sets
- All data sets that currently match specified criteria

Tip: You can schedule a recurring deletion request. For example, you can schedule weekly deletion of all data sets that haven’t been updated in the last 7 days.

When you run the Import and Load Data process, it determines whether a schedule exists for the Delete HCM Data Loader Stage Table Data process. If no schedule exists, then the process creates one to run Delete HCM Data Loader Stage Table Data nightly.

When you delete processed data sets from the HCM Spreadsheet Data Loader stage tables, they’re deleted automatically from the HCM Data Loader stage tables. You can also delete spreadsheet data sets explicitly from the HCM Data Loader.
How Delete HCM Data Loader Stage Table Data Works

The **Delete HCM Data Loader Stage Table Data** process deletes data sets from the HCM Data Loader stage tables. Whenever it runs, the process:

1. Purges automatically from the stage tables any data set that hasn’t been updated in the last 30 days. You can specify a different number of days by setting the **Days to Retain Data Sets Before Deleting** configuration parameter.

2. Reviews the number of data lines still in the stage tables. If the number exceeds a specified maximum value, then data sets may be backed up automatically in archive stage tables. You specify:
   - The maximum number of data lines on the **Maximum Data Lines Before Archiving** configuration parameter.
   - The number of days since a data set was last updated before it can be backed up. Specify this value on the **Days to Retain Data Sets Before Archiving** configuration parameter.

Therefore, a data set is a candidate for automatic backup only if:

- The number of data lines in the stage tables exceeds the value specified on the **Maximum Data Lines Before Archiving** parameter.
- The data set hasn’t been updated in the number of days specified on the **Days to Retain Data Sets Before Archiving** parameter.

The backup starts with the oldest data set and continues until the number of data lines in the stage tables drops below the specified maximum.

Backed-Up Data Sets

These values remain in the stage tables when a data set is backed up:

- The statuses of the data set and business objects
- The record count
- Any error messages

All other data from the data set is moved to the archive stage tables. After a data set is backed up:

- You can’t open error management pages for that data set.
- It doesn’t appear in search results when you search for data sets to purge.
- You can access the data using a custom query only.

You can back up data sets manually by running the **Delete HCM Data Loader Stage Table Data** process with the **Action** parameter set to **Archive**. You can’t create a schedule when the action is **Archive**.

**Related Topics**

- **HCM Data Loader Configuration Parameters**
Example of the Automatic Deletion and Backup of Stage Table Data

When you perform the Delete Stage Table Data task, it runs the Delete HCM Data Loader Stage Table Data process. This process maintains the HCM Data Loader stage tables automatically. This example shows how automatic maintenance of the stage tables works in conjunction with a user request to delete stage table data.

Maintaining the Stage Tables Automatically

The user performs the Delete Stage Table Data task on 9 May 2018 to purge any data sets:

- That weren’t updated in the last 5 days
- Whose Import and Load statuses are both Success

Configuration parameters have their default values, as follows:

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days to Retain Data Sets Before Archiving</td>
<td>5</td>
</tr>
<tr>
<td>Days to Retain Data Sets Before Deleting</td>
<td>30</td>
</tr>
</tbody>
</table>

The following table shows the status of recent import and load processes when the user performs Delete Stage Table Data.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Imported Status</th>
<th>Loaded Status</th>
<th>Last Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Success</td>
<td>Success</td>
<td>9 May 2018</td>
</tr>
<tr>
<td>2</td>
<td>Success</td>
<td>Error</td>
<td>7 May 2018</td>
</tr>
<tr>
<td>3</td>
<td>Success</td>
<td>Success</td>
<td>1 May 2018</td>
</tr>
<tr>
<td>4</td>
<td>Success</td>
<td>Error</td>
<td>28 April 2018</td>
</tr>
<tr>
<td>5</td>
<td>Success</td>
<td>Success</td>
<td>8 April 2018</td>
</tr>
<tr>
<td>6</td>
<td>Success</td>
<td>Error</td>
<td>8 April 2018</td>
</tr>
</tbody>
</table>
These data sets are either purged or backed up as follows:

1. Data sets 3 and 5 are purged by the user’s process, based on the user’s parameters.
2. Data set 6 is purged automatically because it hasn’t been updated in the last 30 days.
3. Data set 4 is backed up automatically. The backup occurs because the number of data lines in the stage tables still exceeds the configured Maximum Data Lines Before Archiving value.

Maintain the HCM Data Loader Stage Tables

This topic explains how to maintain the stage tables for HCM Data Loader by either deleting data sets immediately or scheduling a recurring deletion request.

Delete a Single Data Set on the Import and Load Data Page

Follow these steps to delete a single data set from the stage tables on the Import and Load Data page:

1. On the Tasks panel tab of the Data Exchange work area, select Import and Load Data.
2. Search for and select the data set that you want to delete.
3. In the Search Results section, click the Delete icon for the selected data set.
4. In the Schedule Request dialog box, set the Delete Source File parameter to Yes to delete the source file from the Oracle WebCenter Content server.

Note: You can delete the data from the stage tables without also deleting the source file. If you loaded this data set with the Delete Source File parameter set to Yes, then the source file has already been deleted. In this case, selecting Yes in the Schedule Request dialog box has no effect.

5. Click Submit. The ID of the deletion process appears.
6. Click OK to close the Confirmation dialog box.

Delete Multiple Data Sets on the Delete Stage Table Data Page

You can delete one or more data sets immediately from the stage tables. Alternatively, you can schedule a recurring deletion process. Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select Delete Stage Table Data.
2. On the Delete Stage Table Data page, set the data set criteria to identify the data sets you that you want to delete.
   
   This table describes the main criteria.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days Since Last Updated</td>
<td>Specify the number of days since the data set was last updated. For example, to identify any data set not updated in the last week, set this field to 7. The default value is 30, the maximum value is 60, and the minimum value is 1.</td>
</tr>
<tr>
<td>File Type</td>
<td>Select one of:</td>
</tr>
<tr>
<td>Import Status</td>
<td>Select a status. For example, set to Success to list data sets that imported successfully.</td>
</tr>
</tbody>
</table>

Note: You can delete the data from the stage tables without also deleting the source file. If you loaded this data set with the Delete Source File parameter set to Yes, then the source file has already been deleted. In this case, selecting Yes in the Schedule Request dialog box has no effect.

5. Click Submit. The ID of the deletion process appears.
6. Click OK to close the Confirmation dialog box.
### Best Practices for HCM Data Loader and HCM Spreadsheet Data Loader

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set</td>
<td>Specify a particular data set.</td>
</tr>
<tr>
<td>Created By</td>
<td>Enter the user name of the user who created the data set.</td>
</tr>
<tr>
<td>Load Status</td>
<td>Select a status. For example, set to <strong>Success</strong> to list data sets that loaded successfully.</td>
</tr>
<tr>
<td>Content ID</td>
<td>Enter the Oracle WebCenter Content ID.</td>
</tr>
</tbody>
</table>

#### 3. Click **Show Current Results**.

#### 4. To delete all data sets in the Data Sets table immediately:

a. Click **Schedule Deletion**.

b. On the Parameters tab of the Schedule Request page, the data set criteria from the Delete Stage Table Data page appear. You can’t edit these values on the Schedule Request page.

**Tip:** To change the data set criteria, edit them on the Delete Stage Table Data page, click **Show Current Results**, and click **Schedule Deletion** again. You must click **Show Current Results** before clicking **Schedule Deletion** again. You must click **Show Current Results** before clicking **Schedule Deletion**. Otherwise, the parameters aren’t updated on the Parameters tab of the Schedule Request page.

c. Source files for the data sets are deleted from the Oracle WebCenter Content server by default. To retain the source files, set the **Delete Source File** parameter to **No**.

d. Click **Submit**. The ID of the deletion process appears.

#### 5. To schedule a recurring deletion process for all data sets that match the specified criteria:

a. Click **Schedule Deletion** on the Delete Stage Table Data page.

b. On the Schedule Request page, click the Schedule tab.

c. Set **Run** to **Using a schedule**.

d. Specify the frequency of the deletion process. For example, you could schedule the process to run weekly to delete any data set that hasn't been updated in the last 7 days.

e. Enter start and end dates.

**Note:** If the start date is tomorrow’s date, then the process runs tomorrow and at the specified interval (for example, weekly) until the end date. If the start date is today's date, then the process runs for the first time in one week's time, assuming a weekly frequency.

f. Click **Submit**.

You can review the status of any deletion process by clicking **Review Processes** on the Delete Stage Table Data page. On the Processes page:

- You can see the parameters for any process by clicking the contextual action icon next to the process name. If the process was scheduled, then the Parameters dialog box also includes the schedule details. Scheduled processes are identified by an icon in the **Schedule** column.
• Recurring deletion requests appear on the Processes page with a parent process in *Wait* status. The parent process starts a child process for each scheduled deletion request. The ID of the parent process appears in the **Scheduling Process ID** column.

• The log file for a process includes the request parameters and identifies the data sets that were deleted.

## Delete Spreadsheet Data Sets

On the Delete Stage Table Data page, your search results include spreadsheet data sets if you set **File Type** to **Spreadsheet**. You can delete these data sets from the HCM Data Loader stage tables. But, the spreadsheet data sets remain in the HCM Spreadsheet Data Loader stage tables. So, you can continue to edit and upload these spreadsheet data sets after you have deleted them from the HCM Data Loader stage tables. Such a spreadsheet is reloaded from the HCM Spreadsheet Data Loader stage tables when you next save or upload it. To delete spreadsheet data sets from both sets of stage tables, use the **Delete Spreadsheet Stage Table Data** task.

## Maintain the HCM Spreadsheet Data Loader Stage Tables

This topic explains how to maintain the **stage tables** for HCM Spreadsheet Data Loader. You can either delete data sets immediately or schedule a recurring deletion request.

### Delete Data Sets from the HCM Spreadsheet Data Loader Stage Tables

Follow these steps:

1. On the Tasks panel tab of the Data Exchange work area, select **Delete Spreadsheet Stage Table Data**.
2. On the Delete Spreadsheet Stage Table Data page, set the data set criteria to identify the data sets that you want to delete. This table describes the criteria:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days Since Last Updated</td>
<td>Specify the number of days since the data set was last updated. For example, to identify any data set not updated in the last week, set this field to 7. The default value is 30, the maximum value is 60, and the minimum value is 1.</td>
</tr>
<tr>
<td>Template</td>
<td>Select an HCM Spreadsheet Data Loader template to identify any data set created in spreadsheets generated from the template.</td>
</tr>
<tr>
<td>Data Set</td>
<td>Specify a particular data set. If multiple spreadsheets have been used to load the data set, then they’re all listed.</td>
</tr>
<tr>
<td>Status</td>
<td>Select one of:</td>
</tr>
<tr>
<td></td>
<td>◦ Import completed with errors</td>
</tr>
<tr>
<td></td>
<td>◦ Load completed with errors</td>
</tr>
<tr>
<td></td>
<td>◦ Load successful</td>
</tr>
<tr>
<td></td>
<td>◦ Ready to process</td>
</tr>
<tr>
<td>Last Updated By</td>
<td>Enter the user name of the user who last updated the data set.</td>
</tr>
</tbody>
</table>
3. Click **Show Current Results**.

4. To delete a single data set, select the data set in the Data Sets table and click the **Delete** icon.

5. To delete all data sets in the Data Sets table immediately:
   
   a. Click **Schedule Deletion**.
   
   b. On the Parameters tab of the Schedule Request page, the data set criteria from the Delete Spreadsheet Stage Table Data page appear. You can’t edit these values on the Schedule Request page.

   Tip: To change the data set criteria, edit them on the Delete Spreadsheet Stage Table Data page, click **Show Current Results**, and click **Schedule Deletion** again. You must click **Show Current Results** before clicking **Schedule Deletion**. Otherwise, the criteria aren’t updated on the Parameters tab of the Schedule Request page.

   c. Click **Submit**.

6. To schedule a recurring deletion process for all data sets that match the specified criteria:
   
   a. Click **Schedule Deletion** on the Delete Spreadsheet Stage Table Data page.
   
   b. On the Schedule Request page, click the Schedule tab.
   
   c. Set **Run** to **Using a schedule**.
   
   d. Specify the frequency of the deletion process. For example, you could schedule the process to run weekly to delete any data set that hasn’t been updated in the last 7 days.
   
   e. Enter start and end dates.

   Note: If the start date is tomorrow’s date, then the process runs tomorrow and at the specified frequency (for example, weekly) until the end date. If the start date is today’s date, then the process runs for the first time in one week’s time, assuming a weekly frequency.

   f. Click **Submit**.

You can review the status of any deletion process by clicking **Review Processes** on the Delete Spreadsheet Stage Table Data page. On the Processes page:

- You can see the parameters for any process by clicking the contextual action icon next to the process name. If the process was scheduled, then the Parameters dialog box also includes the schedule details. Scheduled processes are identified by an icon in the **Schedule** column.

- Recurring deletion requests appear on the Processes page with a parent process in **Wait** status. The parent process starts a child process for each scheduled deletion request. The ID of the parent process appears in the **Scheduling Process ID** column.

- The log file for a process includes the request parameters and identifies the data sets that were deleted.

When you delete data sets from the HCM Spreadsheet Data Loader stage tables, they’re also deleted from the HCM Data Loader stage tables.
32 Time, Absence, and Pension Data for Payroll

Overview

You can automate the regular import of time card entries, absence entries, and pension enrollments. You use predefined flows where you specify the interface type and XML file containing the data to load, unless the application pushes the data directly to payroll. For example, recording an absence in Oracle Fusion Absence Management automatically transfers the data to payroll.

This topic covers:
- Loading data options
- Using an XML file to load data

Loading Data Options

The application you use to record time, absences, and pensions determines options for importing data to payroll, as shown in the following table.

<table>
<thead>
<tr>
<th>Source Application</th>
<th>Data Load Options</th>
</tr>
</thead>
</table>
| Time card          | • For Oracle Fusion Time and Labor, submit the Load Time Card Batches process.  
                    | • For a third-party time application, submit the Load Time Card Batches process and load an XML file. |
| Absences           | • For Absence Management, use the Manage Absence Records task to record an absence and transfer it directly to the person’s absence calculation card.  
                    | When you set up the plan in Absence Management, you must select the Transfer absence payment information for payroll processing check box. Find this setting in the Payroll Integration section of the Entries and Balances tab on the Create Absence Plan page.  
                    | • For third-party absence applications, submit the Load Absence Batches process and load an XML file. |
| Pensions           | • Submit the Load Benefit Batches process and load an XML file.  
                    | The availability of this process depends on the implementation for your country or territory. |

Using an XML File to Load Data

To load data by submitting a process to transfer data using an XML file format, do the following steps:

1. Extract your data into an XML file, using the required XML file format and tags.

   For time card entries, if your file is over 2 MB, stage the file on the Oracle WebCenter Content Server and note the Content Id.
2. Select the Submit a Payroll Flow task from the Payroll Checklist work area and select the appropriate process: Load Time Card Batches, Load Absences Batches, or Load Benefits Batches.

3. On the Enter Parameters page, enter a name to identify the flow.

4. In the Interface field, select the option to import an XML file.

5. For time card entries, if your file is over 2 MB, specify the number for the Content ID submission parameter.

6. Optionally, specify a process configuration group.

7. Optionally, schedule the process to transfer the time card entries, absences, or benefit enrollments to payroll.

8. Click the Submit button on the Review page. In the Confirmation window, click the Ok and View Checklist button.

9. On the Task Details tab of the payroll flow, click the Go To Task button for the Upload File task.

10. On the Upload File page in the File field, click the Add button to display the Manage Attachments window.

11. In the Type field, select File.

12. Search for and select your XML file. Click the OK button.

The load batches task starts automatically after the Upload File task completes.

When the process completes successfully, it creates or updates a calculation card for each person included in the XML file. Use the Manage Calculation Cards task in the Payroll Calculation work area to view the new entries on the cards.

Related Topics

- File Format for Importing Time Entries to Payroll

Prerequisite Payroll Setup for Importing Time Entries: Explained

Before you can import time card entries for processing in a payroll run or extracting for a third-party payroll provider, you complete payroll setup tasks. These tasks vary depending on your configuration, specifically whether your enterprise:

- Uses Oracle Fusion Time and Labor or a third-party time provider
- Uses Oracle Fusion Global Payroll or a third-party payroll provider
- Requires workers to submit a time card if their pay is calculated using time entries
The following figure and table lists the tasks that depend on each of these configuration choices.

All configurations create elements for use with time cards. The following table describes the remaining setup tasks and processes that vary based on your configuration.

<table>
<thead>
<tr>
<th>Task</th>
<th>Applies To</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create elements for time card entries</td>
<td>All configurations</td>
</tr>
<tr>
<td>Determine which cost segments workers complete on their time cards</td>
<td>Optional for all configurations</td>
</tr>
<tr>
<td>Create an HCM extract definition of time card elements</td>
<td></td>
</tr>
<tr>
<td>Schedule process to transfer time card entries (optional)</td>
<td></td>
</tr>
<tr>
<td>Notify HR to select Time Card Required field for new hires</td>
<td></td>
</tr>
<tr>
<td>Determine process to transfer time card entries to payroll</td>
<td></td>
</tr>
</tbody>
</table>
Task | Applies To
--- | ---
Record costing overrides at the element entry level by having workers specify the account to charge on their time cards | Third-party time provider

**Note:** If you enable costing overrides, confirm you are using the segments that are enabled for element entry level costing in the Cost Allocation key flexfield.

Create an HCM extract definition of time card elements | Third-party time provider

Determine process to transfer time card entries to payroll:

- Use the Load Time Card Batches process to upload time entries in an XML file to calculation cards
- Use the Payroll Batch Loader to load element entries | Third-party time provider

Schedule process to transfer time card entries | Optional for all configurations

As an example, schedule the Load Time Card Batches process:

- After normal working hours to distribute the load on server
- To run more frequently during time periods when most employees submit their time cards | Optional for all configurations

Create an HCM extract definition that includes time entries | Third-party payroll provider

Notify HR to select Time Card Required field at the Terms or Assignment level when a new hire’s pay calculations will depend on time cards | Optional for all configurations

**Related Topics**

- Schedule Flows
- Extract Components
- Time Card Required Option
Create Elements for Time Card Entries

You create nonrecurring elements to process pay based on time card entries, such as elements for regular, overtime, double-time, and shift pay. Creating a time card element generates all the related elements, balances, formulas, and calculation components. You then transfer the elements to your time provider.

This topic covers the following procedures:

- Creating earnings elements
- Creating calculation components for standard-category elements
- Converting elements for use in time cards
- Setting up area overrides
- Setting up costing overrides

Creating Earnings Elements

The steps for creating a time card element depend on whether the time card template is available for your country. If the template is available, follow the steps in this section. Otherwise, create an earnings element using the Standard category, and specify an hours multiplied by rate calculation rule.

Complete these steps to create an element using the time card template:

1. Create an earnings element on the Manage Elements page of the Payroll Calculation work area.
2. Select a primary classification of standard or supplemental earnings.
3. For Global Payroll, select the Time Card category.
4. Complete the information on the Basic Details page.
5. On the Additional Details page:
   a. Select the calculation units to use in reports.
      Typically, you select time units that match the time units entered on time cards for that element. If you select different units, the application uses 8 hours to convert days to hours.
   b. Optionally, select a default rate to calculate time.
      When calculating the run result for the element entry, the formula uses the default rate unless a rate is entered on the person’s time card.
6. Complete the element eligibility information for the new time element, and its associated retroactive and related elements, such as the result and calculation elements.

Note: If the straight time portion of overtime is reported separately from regular time, create two elements, such as overtime and overtime premium elements. If the regular and straight time portions of overtime are reported together, you might use straight time instead of regular time, and create a separate element for the overtime premium.
Creating Calculation Components for Standard-Category Elements

You can create calculation components for elements created with the Standard category rather than the Time Card category. Complete the following steps for each existing element:

1. Submit the Create Time Card Calculation Components process from the Payroll Checklist or Payroll Administration work area.

   These elements must have a calculation rule of hours multiplied by rate.

2. Complete the element eligibility information for the element and its associated retroactive and its related elements, including the result element, and the element with a suffix of CIR.

3. After you run the process to convert your elements, submit the Compile Formula process in the Manage Payroll Calculations work area. Perform a bulk compile by entering wild cards in the Formula and Formula Type parameters.

Generate Time Card Fields for Your Elements

After creating elements, generate time card fields for them. Complete the processes in this table using the Time and Labor functional area in the Setup and Maintenance work area, Workforce Deployment offering:

<table>
<thead>
<tr>
<th>Process</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate Data Dictionary Time Attributes</td>
<td>Creates dependent payroll attributes for all element input values, such as hours and rate. You must run the Generate Data Dictionary Time Attributes process after making any changes to time elements. Such changes include adding or deleting elements, editing input values, or editing element eligibility records.</td>
</tr>
</tbody>
</table>

⚠️ **Caution:** Failure to run the process might negatively impact the setup of time card fields, the validation of payroll time types, or the transfer of time to payroll.

| Generate Time Card Fields        | Creates time card fields using the data dictionary time attributions for the specified legislative data group. You can use the Manage Time Card Fields task to create time card fields for single and multiple attributes. |

If you are using a third-party time provider, create an HCM extract for the time card elements. The extract includes the element’s mapping ID that you specify in the XML file when you transfer the time entries to payroll.

Setting Up Area Overrides

Some countries or territories create time card elements with area input values for use as overrides. The overrides enable employers to tax employees based on where they work. For example, employees would specify the area information where they worked on temporary assignment while away from their normal work location. These area entries are then included in the time card records transferred to payroll by the Load Time Card Batches process.

Setting Up Costing Overrides

You can specify additional attributes in Time and Labor to enter costing segment values on time cards. The segments must match the segments that you can enter on element entries. Use the Manage Element Entries task in the Payroll Calculation
and search for a person’s record. View the available segments on the Costing tab of the Manage Person Details page. Costing is defined on the element eligibility record of the results element. When you transfer time entries, the transfer process displays the costing on the calculation element.

As an example, the structure of your cost allocation key flexfield might specify that the department segment is entered at the element entry level. You could specify this additional attribute on the time card. Your employees could then specify the department to charge for overtime hours worked while on loan to a different department. After you transfer the time entries, the payroll calculation uses the department specified for the overtime hours to derive the costing results.

To view the results of the costing overrides transferred to payroll, do the following:

1. On the Accounting Distribution work area, select Search Person.
2. On the Search Person page, search for a person.
3. Click the Actions down arrow and select View Person Process Results in the Process Results option.
4. Go into a person’s statement of earnings.
5. Select Costing Results.

Related Topics
- Processing Time Entries in Payroll
- Time Card Required Option

Importing Time Card Entries to Payroll: Procedure

Import time card entries to payroll by submitting the Load Time Card Batches process from the Payroll Checklist or Payroll Administration work areas. The process creates a new calculation card or updates an existing card for each person with time entries included in the batch.

You use the same process to import time entries from Oracle Fusion Time and Labor and from a third-party time provider. Submitting the time entries for a third-party requires an additional step of creating an XML file that contains the time entries. This topic covers both procedures:

- Importing time entries from Time and Labor
- Importing time entries from a third-party time provider

Importing Time Entries from Time and Labor

Complete the following steps to import your time entries

1. On the Payroll Checklist work area, Tasks panel tab, click Submit a Payroll Flow.
   a. On the Submit a Payroll Flow: Select Flow Pattern page, select the legislative data group.
   b. In the Flow Pattern section, select Load Time Card Batches.
2. Click Next.
3. On the Submit a Payroll Flow: Enter Parameters page, in the Payroll Flow field, enter a name to identify the flow.
4. In the Interface Type field, search for and select ORACLE FUSION TIME AND LABOR.
5. Optionally, specify a process configuration group. Click Next.

>Note: If you don’t need to connect, schedule, or review the flow parameters, you can skip these pages and submit the flow by clicking the Submit button.
6. On the Enter Flow Interaction page, complete the optional parameters. Click Next.
7. On the Schedule page, complete the optional parameters. Click Next.
9. On the Payroll Flow page, Task Details tab, you should see a green check mark in the Upload File row, Task Type column. If not, on the toolbar, click the Refresh icon intermittently until you do.
10. Right-click the Upload File row and select Mark as Complete.

You don’t need to upload a file for Time and Labor. The Upload task is only used to upload an XML file from a third-party time provider.

The View Person Process Results page lists the time card IDs included in the batch of transferred time cards. View the time entries for each person’s time included in the batch using the Manage Calculation Cards task in the Payroll Calculation work area.

Importing Time Entries from a Third-Party Provider

Complete the following steps to transfer your time card entries from a third-party provider:

1. On the Payroll Checklist work area, Tasks panel tab, click Submit a Payroll Flow.
2. On the Submit a Payroll Flow: Select Flow Pattern page, select the legislative data group.
3. In the Flow Pattern section, select Load Time Card Batches.
4. Click Next.
5. On the Submit a Payroll Flow: Enter Parameters page, in the Payroll Flow field, enter a name to identify the flow.
6. In the Interface Type field, search for and select Import Time XML.
7. Optionally, specify a process configuration group.
8. If your file is over 2 MB, specify the number for the Content ID submission parameter.
9. Click Next.
10. On the Enter Flow Interaction page, complete the optional parameters. Click Next.
11. On the Schedule page, complete the optional parameters. Click Next.
12. Click the Submit button on the Review page. In the Confirmation window, click the Ok and View Checklist button.
13. If your file is over 2 MB and you have already provided the Content ID during submissions, skip the Upload File task.

To skip the task, right-click on the row and select Mark as Complete. The load batches tasks starts automatically.

If not, then follow the below steps to upload the XML file:

a. On the Payroll Flow page, Task Details tab, you should see a green check mark in the Upload File row, Task Type column. If not, on the toolbar, click the Refresh icon intermittently until you do.
b. Click the Go to Task icon.
c. On the Upload File page in the File field, click Add to display the Manage Attachments dialog box.
d. In the Type field, select File.
e. Search for and select your XML file. Click OK.
f. On the Upload File page, click Done to complete the submission and initiate the time data transfer.

The load batches task starts automatically after the Upload File task completes.

The View Person Process Results page lists the time card IDs included in the batch of transferred time cards. View the time entries for each person’s time included in the batch using the Manage Calculation Cards task in the Payroll Calculation work area. Entries transferred from third-party providers only display on the calculation cards, and not in the Oracle Fusion Time and Labor time cards.
File Format for Importing Time Entries

You import time entries from a third-party provider by submitting the Load Time Card Batches process from the Payroll Checklist or Payroll Administration work areas. When you submit the process, you specify the batch XML file that includes your time entries. This topic explains the XML file format and XML tags you must use in the file.

XML File Format for Importing Time Entries

When you create a file to transfer time card entries to payroll, use the following structure.

```xml
<TIME_CARD_LIST>
  <TIME_CARD>..
    <ACTION>
      <TIME_CARD_ID>
        <ACTION>
          <TIME_CARD_ID>
            <MAPPING_NAME>
              <LDG_ID>
                <LDG_NAME>
                  <HR_TERM_ID>
                    <TERM_NUMBER>
                      <HR_ASSIGNMENT_ID>
                        <ASSIGNMENT_NUMBER>
                          <LEGAL_EMPLOYER_ID>
                            <LEGAL_EMPLOYER_NAME>
                              <TIME_CARD_START>
                                <TIME_CARD_END>
                                  <TIME_ITEM_LIST>..
                                    <TIME_ITEM>
                                      <TIME_TYPE>
                                        {<PAYMENT_RATE_ID>
                                          <PAYMENT_RATE_NAME>
                                            <RATE_AMOUNT>
                                              <PERIODICITY>
                                                <FACTOR>
                                                  <AMOUNT>
                                                    <PERIODICITY>
                                              }
                                      </PAYMENT_RATE_ID>
                                      <TIME_UNIT>
                                        <TIME_UOM>
                                          <TIME_ITEM_START>
                                            <TIME_ITEM_END>
                                              <COST_SEGMENTS>
                                                <SEGMENT1..30>
                                              </COST_SEGMENTS>
                                              <PROPERTIES_LIST>..
                                              <PROPERTY_ITEM>
                                            </TIME_ITEM_END>
                                          </TIME_ITEM_START>
                                        </TIME_UOM>
                                      </PAYMENT_RATE_ID>
                                    </TIME_ITEM_LIST>
                                  </TIME_CARD_END>
                                </TIME_CARD_START>
                              </LEGAL_EMPLOYER_NAME>
                            </LEGAL_EMPLOYER_ID>
                          </ASSIGNMENT_NUMBER>
                        </HR_ASSIGNMENT_ID>
                      </TERM_NUMBER>
                    </HR_TERM_ID>
                  </LDG_NAME>
                </LDG_ID>
              <HR_MAPPING_NAME>
                <LDG_ID>
                  <LDG_NAME>
                    <HR_TERM_ID>
                      <TERM_NUMBER>
                        <HR_ASSIGNMENT_ID>
                          <ASSIGNMENT_NUMBER>
                            <LEGAL_EMPLOYER_ID>
                              <LEGAL_EMPLOYER_NAME>
                                <TIME_CARD_START>
                                  <TIME_CARD_END>
                                    <TIME_ITEM_LIST>..
                                      <TIME_ITEM>
                                        <TIME_TYPE>
                                          {<PAYMENT_RATE_ID>
                                            <PAYMENT_RATE_NAME>
                                              <RATE_AMOUNT>
                                                <PERIODICITY>
                                                  <FACTOR>
                                                    <AMOUNT>
                                                      <PERIODICITY>
                                                }
                                        </PAYMENT_RATE_ID>
                                        <TIME_UNIT>
                                          <TIME_UOM>
                                            <TIME_ITEM_START>
                                              <TIME_ITEM_END>
                                                <COST_SEGMENTS>
                                                  <SEGMENT1..30>
                                                </COST_SEGMENTS>
                                                <PROPERTIES_LIST>..
                                                <PROPERTY_ITEM>
                                              </TIME_ITEM_END>
                                            </TIME_ITEM_START>
                                          </TIME_UOM>
                                        </PAYMENT_RATE_ID>
                                      </TIME_ITEM_LIST>
                                    </TIME_CARD_END>
                                  </TIME_CARD_START>
                                </LEGAL_EMPLOYER_NAME>
                            </LEGAL_EMPLOYER_ID>
                          </ASSIGNMENT_NUMBER>
                        </HR_ASSIGNMENT_ID>
                      </TERM_NUMBER>
                    </HR_TERM_ID>
                  </LDG_NAME>
                </LDG_ID>
              </HR_MAPPING_NAME>
            </MAPPING_NAME>
          </TIME_CARD_ID>
        </ACTION>
      </TIME_CARD_ID>
    </ACTION>
  </TIME_CARD>..
</TIME_CARD_LIST>
```
XML Tags

This table describes the purpose of the tags used in the XML file.

<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME_CARD_LIST</td>
<td>Parent tag that contains a set of time cards.</td>
</tr>
<tr>
<td>TIME_CARD</td>
<td>Object that contains the information about a specific time card.</td>
</tr>
<tr>
<td>ACTION</td>
<td>Action to perform, such as CREATE, REMOVE, MODIFY.</td>
</tr>
<tr>
<td>TIME_CARD_ID</td>
<td>Unique identifier for this time card.</td>
</tr>
<tr>
<td>MAPPING_ID</td>
<td>Identifier for the payroll component definition.</td>
</tr>
<tr>
<td>MAPPING_NAME</td>
<td>Name used for the mapping.</td>
</tr>
<tr>
<td>LDG_NAME</td>
<td>Name of the legislative data group (LDG) for this record.</td>
</tr>
<tr>
<td>LDG_ID</td>
<td>Identifier for the LDG for this record.</td>
</tr>
<tr>
<td>TERM_NUMBER</td>
<td>Number that identifies the term for the time entry.</td>
</tr>
<tr>
<td>ASSIGNMENT_NUMBER</td>
<td>Number that identifies the assignment for the time entry.</td>
</tr>
<tr>
<td>TIME_CARD_START</td>
<td>Start date of the time card.</td>
</tr>
<tr>
<td>XML Tag</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>TIME_CARD_END</td>
<td>End date of the time card.</td>
</tr>
<tr>
<td>TIME_ITEM_LIST</td>
<td>Tag that contains a set of time items.</td>
</tr>
<tr>
<td>TIME_ITEM</td>
<td>Object that contains information about a specific hour item.</td>
</tr>
<tr>
<td>TIME_TYPE</td>
<td>Name supplied by the time application that maps to the payroll element and calculation component.</td>
</tr>
<tr>
<td>PAYMENT_RATE_ID</td>
<td>Identifier for the rate definition used to calculate the payment amount.</td>
</tr>
<tr>
<td>PAYMENT_RATE_NAME</td>
<td>Name of the rate definition used to calculate the payment amount.</td>
</tr>
<tr>
<td>RATE_AMOUNT</td>
<td>Actual rate used to calculate the payroll amount.</td>
</tr>
<tr>
<td>AMOUNT</td>
<td>Flat amount used to calculate the rate based on periodicity.</td>
</tr>
<tr>
<td>PERIODICITY</td>
<td>Frequency that determines the rate value, used with amount or rate amount.</td>
</tr>
<tr>
<td>FACTOR</td>
<td>Multiplier applied to the derived rate to calculate the payment amount.</td>
</tr>
<tr>
<td>TIME_UNIT</td>
<td>Number of units for the Unit of Measure specified in TIME_UOM. For example, if the UOM is hours, 8 units is 8 hours worked.</td>
</tr>
<tr>
<td>TIME_UOM</td>
<td>Unit of measure for specifying time unit, such as hours.</td>
</tr>
<tr>
<td>TIME_ITEM_START</td>
<td>Start time for the time entry.</td>
</tr>
<tr>
<td>TIME_ITEM_END</td>
<td>Ending time for the time entry.</td>
</tr>
<tr>
<td>COST_SEGMENTS</td>
<td>List of the costing segments.</td>
</tr>
<tr>
<td>PROPERTY_LIST</td>
<td>Set of properties for the time item.</td>
</tr>
<tr>
<td>PROPERTY_ITEM</td>
<td>Additional information that is captured. For example, a value definition for the property item State would return State and the name of the State.</td>
</tr>
<tr>
<td>NAME</td>
<td>Name of a property for the time item.</td>
</tr>
<tr>
<td>VALUE</td>
<td>Value of a property for the time item.</td>
</tr>
</tbody>
</table>
# Integrating Absence Management with Global Payroll: Procedure

Use Oracle Fusion Absence Management to set up and enroll persons in absence plans. Transfer absence information to Oracle Fusion Global Payroll for processing and paying absences.

Integrating Absence Management with payroll requires that you complete steps in different work areas for Absence Management, Human Resources, and payroll. The following table displays the sequence of tasks to perform in each work area.

<table>
<thead>
<tr>
<th>Steps</th>
<th>Absences work area</th>
<th>Person Management work area</th>
<th>Set Up and Maintenance work area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an absence plan, such as an Accrual, Qualification, or No Entitlement plan.</td>
<td>Manage Absence Plans task</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Create absence types based on predefined absence patterns and associate them to the absence plans.</td>
<td>Manage Absence Types task</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Create rate definitions to use in calculating accrual and liability balances.</td>
<td>NA</td>
<td>NA</td>
<td>Manage Rate Definitions task</td>
</tr>
<tr>
<td>Create absence elements.</td>
<td>NA</td>
<td>NA</td>
<td>Manage Elements task</td>
</tr>
<tr>
<td>Complete the following payroll integration information:</td>
<td>Manage Absence Plans task</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>• Select the Transfer absence payment information for payroll processing check box.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Select the element for the plan in the Element field. The element links the absence plan to the calculation component shown on the person's calculation card</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Optionally, specify rates. These rates override rates specified elsewhere.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Find these settings in the Payroll Integration section of the Entries and Balances tab on the Create Absence Plan page.

| Enroll persons in the absence plan: | NA | Manage Absence Records task | NA |
| Add the plan and provide the entitlement date information. |  |  |  |
Steps | Absences work area | Person Management work area | Set Up and Maintenance work area
--- | --- | --- | ---
• For accrual plans, select the Run Accruals for Selected Plan task action.

After setting up the information, you can record an absence using the Manage Absence Records task in the Person Management work area. When the absence is approved, the daily and summary breakdown information is automatically transferred to payroll.

You can then process the payroll that includes these absence entries and view the resulting absence balances on the person’s Statement of Earnings. After you process and archive payments, you can view the resulting absence balances on the person’s payslip.

**Related Topics**
- How Many Salary Bases to Create
- Rates Used to Calculate Absences in Payroll

### Define Payroll Elements to Process Absences

You define elements to calculate and process absence payments in Oracle. When you define an absence element, your responses to the element template questions determine which elements, balances, formulas, and calculation components the template generates.

Defining an absence element involves the following steps:
- Defining an absence element
- Completing absence detail questions
- Completing accrual liability and balance payment questions
- Completing absence payment questions
- Submitting the element
- Defining element eligibility records and cost distributions

### Defining an Absence Element

Define an absence element, selecting a primary classification of Absence, and a secondary classification. Typically, the predefined values include vacation, maternity, sickness, and other. Use the Manage Elements task in the Payroll Calculation work area.

### Completing Absence Detail Questions

The questions you complete in the Absence Details section determine which subsequent questions the template displays. You enter the following information in the Absence Details section:

1. Specify the calculation units to use when reporting the absence, for example that is shown on the payslip, and statement of earnings. Typically, you select Days or Hours for your reports that correspond to the units for your absence plan. When creating an absence element, select the work calculation rule to calculate the absence rate.
2. Select the absence information to transfer to payroll based on the type of absence management plan.
Completing Accrual Liability and Balance Payment Questions

If you transfer accrual balances, complete the questions shown in the following table.

<table>
<thead>
<tr>
<th>Question</th>
<th>Steps</th>
</tr>
</thead>
</table>
| Calculate absence liability? | 1. Select Yes, if you calculate liability.  
2. Select a liability rate.  
Usually the rate is the same as the absence payment rate. You might select a different rate when estimating liability for billing purposes. |
| Does this plan enable balance payments when enrollment ends? | 1. Select Yes to configure a final disbursement element and to maintain balances for the disbursement hours and payments.  
2. Optionally, select a rate to use for the calculation. |
| Does this plan enable partial payment of balance? | 1. Select Yes to configure a discretionary disbursement element and to maintain balances for disbursement hours and payments.  
2. Optionally, select a rate to use for the calculation. |

Completing Absence Payment Questions

Complete the following questions:

1. Select a method to reduce regular earnings if employees don’t complete a time card, or the time card entries aren’t used as a basis for calculating pay:
   - Reduce regular earnings by absence payment (entitlement payment balance)
   - Select rate to determine absence deduction amount (entitlement deduction balance)

   You might select one of the following:
   - The **Reduce regular earnings** option to reduce regular earnings by the absence payment. This means that the employee is paid the same net amount as if they weren’t absent.
   - The **Select rate to determine deduction amount** option when the employee is not due to be paid for the absence at the same rate as their regular earnings. In this case, the absence deduction rate that you select will be a rate that deducts 100% of the regular earnings. However, the absence payment rate would be a different rate, for example 50%.

2. Optionally, select a rate to calculate the absence payment.
If you have standard earnings and absence elements in the same payroll run that reduce regular earnings, the payroll calculation reduces earnings in this sequence:

- a. Using absence element entries
- b. Using any standard earnings elements that reduce regular earnings

The salary balance isn’t reduced beyond zero.

3. You can now ensure that absence entitlement payments are made to employees after terminations. Use the Does this plan enable entitlement payments after termination? question on the absence element template to set the entitlement element to Final Close.

Example: Amelia is due to be paid maternity payments after her termination. Select Yes in this newly-added question to set the latest entry date of the entitlement elements to final close. With this, the absence entitlement payments are made to Amelia after her termination.

In addition to this, you will need to do the following:

- When Amelia is terminated, change her employment assignment status to Process When Earning.
- Set the TERM_INCLUDE_PR_LEVEL action parameter to Y, so that payroll relationship level entries are considered for processing.

Submitting the Element

When you submit the element, the template automatically configures a base pay element, balances, formulas, and calculation components.

The template also configures additional elements, depending on the options selected in the template to transfer absence information, as shown in the following table.

<table>
<thead>
<tr>
<th>Type of Absence Information to Transfer</th>
<th>Optional Balance Payments Selected</th>
<th>Additional Elements Configured</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accrual Balances</td>
<td>• Discretionary Disbursements</td>
<td>• Accrual</td>
</tr>
<tr>
<td></td>
<td>• Final Disbursements</td>
<td>• Discretionary Disbursement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Final Disbursement</td>
</tr>
<tr>
<td>Accrual Balances and Absences</td>
<td>• Discretionary Disbursements</td>
<td>• Accrual</td>
</tr>
<tr>
<td></td>
<td>• Final Disbursements</td>
<td>• Entitlement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Discretionary Disbursement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Final Disbursement</td>
</tr>
<tr>
<td>Qualification Absences</td>
<td>none</td>
<td>Entitlement</td>
</tr>
<tr>
<td>No Entitlement Absences</td>
<td>none</td>
<td>Entitlement</td>
</tr>
</tbody>
</table>

Defining Element Eligibility Records and Cost Distributions

Define element eligibility records for all the elements generated by the template, for example for your accrual, entitlement, discretionary and final disbursement elements.
If your enterprise calculates cost distributions, specify costing for all the element eligibility records. For example, for an accrual element, you do the following steps:

1. Define element eligibility records for the accrual, accrual results, accrual retroactive, and accrual retroactive results elements.
2. Specify costing for the accrual results and retroactive results elements.

The costing process would cost the change in the liability balance since the last payroll period, debit the expense account and credit the liability account.

Related Topics
- Costing of Elements Options
- Define Payroll Elements for an Absence Accrual Plan
- Rates Used to Calculate Absences in Payroll

Importing Absence Entries to Payroll: Procedure

You can import worker absences from Oracle Fusion Absence Management or another absence application to Oracle Fusion Global Payroll to process absence payments. When you set up an absence plan, you associate an absence element to transfer the absence information to a person's calculation card. Processing absence information for payroll includes the following aspects.

- Importing absence entries
- Processing absence entries
- Validating absence results
- Correcting absence results

Importing Absence Entries from Absence Management

Recording an absence in Absence Management transfers a summary record and a daily breakdown for each day the person is absent to the person's calculation card. The summary record contains information such as the start and end date of the absence, the absence rate, the units. The daily breakdown shows the absence date and any factor to be applied, such as a percentage to use when calculating the absence payment.

Importing Absence Entries from a Third-Party

If you use a third-party absence application, you create an XML file for the absence entries and transfer it to payroll using the Load Absence Batches process. Complete the following steps:

1. Extract your absence data into an XML file. You must use the required file format and XML tags.
2. Use the Submit a Payroll Flow task to submit the Load Absence Batches flow from the Payroll Administration or Payroll Checklist work areas.
3. On the Enter Parameters page, enter a name to identify the flow.
4. In the Interface field, select Import Absence XML.
5. Optionally, specify a process configuration group.
6. Optionally, complete the parameters on the Flow Interaction page.
7. Optionally, complete the parameters on the Scheduling page.
8. Click the Submit button on the Review page. In the Confirmation window, click the Ok and View Checklist button.
9. On the Task Details tab of the payroll flow, click the Go To Task button for the Upload File task.
10. On the Upload File page in the File field, click the Add button to display the Manage Attachments window.
11. In the Type field, select File.
12. Search for and select your XML file. Click the OK button.

The load batches task starts automatically after the Upload File task completes.

The View Person Process Results page lists the absence IDs included in the batch of transferred absences. View the absence entries for each person included in the batch using the Manage Calculation Cards task in the Payroll Calculation work area.

### Processing Absence Entries

Transferring the absence information to a calculation card creates an element entry for the element associated to the absence plan. The payroll run processes this entry using the formula attached to the element and values from the calculation card. The resulting values are then passed back to the element entry and stored as run results and balances.

### Validating Absence Results

You can verify the results of absences after calculating run results by using payroll reports or the statement of earnings. After archiving payment results, validate absence information on the Payroll Register Report or the person’s payslip.

Review absence results on the following payroll reports:

- Element Result Report
- Gross-to-Net Report
- Payroll Activity Report
- Payroll Balance Report
- Payroll Register Report

Use the View Person Process Results task in the Payroll Calculation work area or click the person’s record on the Person Process Results page of the payroll calculation flow to go to the person’s statement of earnings. The following table lists the absence information displayed on the statement of earnings.

<table>
<thead>
<tr>
<th>Statement of Earnings Sections</th>
<th>Information Displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absences Accrual tab in the Quick Reference Summary</td>
<td>Accrual balances for accrual days and accrual hours</td>
</tr>
</tbody>
</table>
| Absences | • Total for the entitlement, discretionary disbursement, final disbursement payment  
• Absence deductions made for that period |

Use the View Payslip task to view the person’s absence information in the Person Management work area. The following table lists the absence information displayed on the payslip.

<table>
<thead>
<tr>
<th>Payslip Sections</th>
<th>Information Displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Total amount for the following balances: entitlement payment, entitlement deduction, final disbursement payment, and discretionary disbursement payment</td>
</tr>
</tbody>
</table>
## Payslip Sections

<table>
<thead>
<tr>
<th>Section</th>
<th>Information Displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absence</td>
<td>Payment balances that contribute to the total absence balance shown in the Summary region</td>
</tr>
<tr>
<td>Absence Accrual</td>
<td>Balances for accrual days and accrual hours</td>
</tr>
</tbody>
</table>

> **Note:** Subregions display hours and rate, and days and rate details for these balances

For accrual liabilities, the Employer Charges section of the Statement of Earnings page and payslip display the difference for liability balance since the last payroll.

### Correcting Absence Results

You can continue to transfer updated absence entries to a person’s calculation card until the payroll calculation starts. Any further updates are then processed as retroactive absences entries in the following payroll run.

### File Format for Importing Absence Entries

When you submit the Load Absence Batches process, you specify the attachment for the XML file that contains the absence data. This topic explains the XML file format and XML tags you must use in the file.

You submit the Load Absence Batches process from the Payroll Administration work area. The process creates a new calculation card or updates an existing card for each worker whose absence information is transferred.

### XML File Format for Importing Absence Information to Payroll

When you create a file to transfer absence information to payroll, use the following format.

```xml
<ABSENCE_LIST>
  <ABSENCE>
    <ABSENCE_TYPE>
      <ACTION>
        <ABSENCE_ID>
          <MAPPING_ID>
            <MAPPING_NAME>
              <LDG_ID>
                <LDG_NAME>
                  <HR_TERM_ID>
                    <TERM_NUMBER>
                      <HR_ASSIGNMENT_ID>
                        <ASSIGNMENT_NUMBER>
                          <ABSENCE_RATE_ID>
                            <ABSENCE_RATE_NAME>
                              <ABSENCE_UNIT>
                                <ABSENCE_UOM>
                                  <ADJUSTMENT_UNIT>
                                    <FACTOR>
                                      <CALCULATION_DATE>
                                        <PERIODICITY>
                                          <ABSENCE_START>
                                            <ABSENCE_END>
  </ABSENCE>
</ABSENCE_LIST>
```
XML Tags

This table describes the purpose of the tags used in the XML file.

<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSENCE_LIST</td>
<td>Outermost tag that contains a set of absences.</td>
</tr>
<tr>
<td>ABSENCE</td>
<td>Tag containing information about a particular absence.</td>
</tr>
<tr>
<td>ABSENCE_TYPE</td>
<td>Type of absence that is being transferred to payroll, such as accrual, accrual with entitlement, or entitlement.</td>
</tr>
<tr>
<td>ACTION</td>
<td>Type of action that will be performed, such as CREATE, REMOVE, and MODIFY.</td>
</tr>
<tr>
<td>ABSENCE_ID</td>
<td>Unique identifier for the absence from the source application. Never use the same ID twice to identify another absence.</td>
</tr>
<tr>
<td>MAPPING_ID</td>
<td>Identifier for the payroll component definition, which is used to create the absence in payroll.</td>
</tr>
<tr>
<td>MAPPING_NAME</td>
<td>Name used for the mapping.</td>
</tr>
<tr>
<td>LDG_ID</td>
<td>ID of the legislative data group associated with the record.</td>
</tr>
<tr>
<td>LDG_NAME</td>
<td>Name of the legislative data group associated with the record.</td>
</tr>
<tr>
<td>HR_TERM_ID</td>
<td>Unique ID for the HR period.</td>
</tr>
<tr>
<td>TERM_NUMBER</td>
<td>Number that identifies the employment periods for the absence.</td>
</tr>
<tr>
<td>HR_ASSIGNMENT_ID</td>
<td>This is the HR Assignments unique ID.</td>
</tr>
</tbody>
</table>

You can provide either the TERM_NUMBER or the HR_TERM_ID. If you provide the TERM_NUMBER then you must also provide the legal employer details.

You can provide either the ASSIGNMENT_NUMBER or the HR_ASSIGNMENT_ID. If you provide the ASSIGNMENT_NUMBER then you must also provide the legal employer details.
<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSIGNMENT_NUMBER</td>
<td>Number that identifies the employment assignment for the absence.</td>
</tr>
<tr>
<td>ABSENCE_RATE_ID</td>
<td>Unique identifier for the absence rate.</td>
</tr>
<tr>
<td>ABSENCE_RATE_NAME</td>
<td>Name of the rate used to calculate the payment amount.</td>
</tr>
<tr>
<td>ABSENCE_UNIT</td>
<td>Unit of time in which the absence is recorded.</td>
</tr>
<tr>
<td>ABSENCE_UOM</td>
<td>Unit of measure being used for the absence (for example, days, hours or weeks).</td>
</tr>
<tr>
<td>ADJUSTMENT_UNIT</td>
<td>Unit of time in which an adjustment is being made to the absence.</td>
</tr>
<tr>
<td>FACTOR</td>
<td>Factor that is used in the calculation of the absence.</td>
</tr>
<tr>
<td>CALCULATION_DATE</td>
<td>Date used for payroll calculations, such as the payment calculation for maternity leave based on the baby’s due date.</td>
</tr>
<tr>
<td>PERIODICITY</td>
<td>Used with the amount or rate, the periodicity is the frequency that determines the absence rate.</td>
</tr>
<tr>
<td>ABSENCE_START</td>
<td>Date the absence started.</td>
</tr>
<tr>
<td>ABSENCE_END</td>
<td>Date the absence ended.</td>
</tr>
<tr>
<td>ABSENCE_DATE_LIST</td>
<td>List of dates in which the absence occurred.</td>
</tr>
<tr>
<td>ABSENCE_DATE</td>
<td>Date the absence is being reported.</td>
</tr>
<tr>
<td>LEAVE_DATE</td>
<td>Date on which the leave of absence occurred.</td>
</tr>
<tr>
<td>ACCRUED_DATE</td>
<td>Date on which the absence was accrued.</td>
</tr>
<tr>
<td>OVERRIDING_FACTOR</td>
<td>Factor that is being used to override the calculation of the absence.</td>
</tr>
<tr>
<td>OVERRIDING_RATE_ID</td>
<td>Unique identifier for the rate being used to override the absence.</td>
</tr>
<tr>
<td>OVERRIDING_RATE_NAME</td>
<td>Name of the overriding rate that will be used to calculate the absence.</td>
</tr>
<tr>
<td>OVERRIDING_UOM</td>
<td>Unit of measure being used to override the absence (for example, days, hours or weeks).</td>
</tr>
<tr>
<td>OVERRIDING_UNIT</td>
<td>Unit of time in which an override is being made to the absence.</td>
</tr>
</tbody>
</table>
Running the Load Benefit Batches Process: Procedure

Use the Load Benefit Batches process to transfer pension deduction information from a benefits application to benefits and pensions calculation cards for payroll processing.

Submitting the Process

You can submit the process from the Payroll Checklist or Payroll Administration work areas, or you can add it to a payroll flow pattern so that it runs as part of your regular payroll flow.

To submit the process:

1. Extract your pension data into an XML file, which must use the required file format and XML tags.
2. Use the Submit a Payroll Flow task to submit the Load Benefit Batches flow from the Payroll Administration or Payroll Checklist work areas.
3. On the Enter Parameters page, enter a name to identify the flow.
4. In the Interface field, select Import Benefit XML.
5. Optionally, specify a process configuration group.
6. Optionally, complete the parameters on the Flow Interaction page.
7. Optionally, complete the parameters on the Scheduling page.
8. Click the Submit button on the Review page. In the Confirmation window, click the Ok and View Checklist button.
9. On the Task Details tab of the payroll flow, click the Go To Task button for the Upload File task.
10. On the Upload File page in the File field, click the Add button to display the Manage Attachments window.
11. In the Type field, select File.
12. Search for and select your XML file. Click the OK button.

The load batches task starts automatically after the Upload File task completes.

Resolving Transfer Errors

When you load a benefits batch, the application validates the entries to confirm that the worker isn’t terminated and is eligible for the deduction. The application rejects entries for any date beyond the worker’s termination date.

If the process ends in error, you can roll it back, resolve the error in the source application, and resubmit the Load Benefit Batches process.

Viewing and Correcting Entries

When the process completes successfully, it creates or updates a calculation card for each person included in the XML file. Use the Manage Calculation Cards task in the Payroll Calculation work area to view the new entries on the cards.

You can enter or update the following values on the cards, if required:

- Payee
- Reference Number
- Employee Additional Contribution

The other values are view-only and must be maintained in the source application.
Related Topics

- Considerations to Enter Calculation Values for Pensions

File Format for Importing Pension Deductions

When you submit the Load Benefit Batches process, you specify the attachment for the XML file that contains the benefit data. This topic explains the XML file format and XML tags you must use in the file. You submit the Load Benefit Batches process from the Payroll Checklist or Payroll Administration work areas. The process creates a new calculation card or updates an existing card for each worker whose pension information is transferred.

XML File Format for Importing Pension Deductions to Payroll

When you create a file to transfer pension deduction information to payroll, use the following format.

```xml
<BENEFIT_LIST>
  <BENEFIT>...
    <ACTION>
      <BENEFIT_ID>
        <MAPPING_ID>
          <LDG_ID>
            <LDG_NAME>
              { ...
                <HR_TERM_ID>
                  <TERM_NUMBER>
                <HR_ASSIGNMENT_ID>
                  <ASSIGNMENT_NUMBER>
              }
              <LEGAL_EMPLOYER_ID>
              <LEGAL_EMPLOYER_NAME>
              <BENEFIT_START>
                <BENEFIT_END>
                  { ...
                    <BENEFIT_RATE_ID>
                      <BENEFIT_RATE_NAME>
                      <AMOUNT>
                      <PERIODICITY>
                      <BENEFIT_MAX_ELECTION>
                      <BENEFIT_REF_NUMBER>
                  }
                </BENEFIT_END>
              } ...
            </LDG_NAME>
          </LDG_ID>
        </MAPPING_ID>
      </BENEFIT_ID>
    </ACTION>
  </BENEFIT>
</BENEFIT_LIST>
```

XML Tags

This table describes the purpose of the tags used in the XML file.

<table>
<thead>
<tr>
<th>XML Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BENEFIT_LIST</td>
<td>Outermost tag that contains a set of benefits.</td>
</tr>
<tr>
<td>XML Tag</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BENEFIT</td>
<td>Tag containing information about a particular benefit.</td>
</tr>
<tr>
<td>ACTION</td>
<td>The type of action that will be performed, such as CREATE, REMOVE, and MODIFY.</td>
</tr>
<tr>
<td>BENEFIT_ID</td>
<td>Unique identifier for the benefit from the source application. Never use the same ID twice to identify another benefit.</td>
</tr>
<tr>
<td>MAPPING_ID</td>
<td>Identifier for the payroll component definition, which is used to create the benefit in payroll.</td>
</tr>
<tr>
<td>LDG_ID</td>
<td>ID of the legislative data group associated with the record.</td>
</tr>
<tr>
<td>LDG_NAME</td>
<td>Name of the legislative data group associated with the record.</td>
</tr>
<tr>
<td>HR_TERM_ID</td>
<td>Unique ID for the employment period.</td>
</tr>
<tr>
<td>TERM_NUMBER</td>
<td>Number that identifies the employment period for the pension deduction.</td>
</tr>
<tr>
<td>HR_ASSIGNMENT_ID</td>
<td>Unique ID for the assignment.</td>
</tr>
<tr>
<td>ASSIGNMENT_NUMBER</td>
<td>Number that identifies the employment assignment for the pension deduction.</td>
</tr>
<tr>
<td>LEGAL_EMPLOYER_ID</td>
<td>ID of the legal employer name that the assignment belongs to.</td>
</tr>
<tr>
<td>LEGAL_EMPLOYER_NAME</td>
<td>Legal employer name that the assignment belongs to.</td>
</tr>
<tr>
<td>BENEFIT_START</td>
<td>Start date of the benefit.</td>
</tr>
<tr>
<td>BENEFIT_END</td>
<td>End date of the benefit.</td>
</tr>
<tr>
<td>BENEFIT_RATE_ID</td>
<td>ID of the rate that will be used to calculate the payment amount.</td>
</tr>
<tr>
<td>BENEFIT_RATE_NAME</td>
<td>Name of the rate that will be used to calculate the payment amount.</td>
</tr>
<tr>
<td>AMOUNT</td>
<td>Amount that is used to calculate the rate using the periodicity.</td>
</tr>
<tr>
<td>PERIODICITY</td>
<td>Used with the amount or rate, the periodicity is the frequency that determines the rate value.</td>
</tr>
<tr>
<td>BENEFIT_MAX_ELECTION</td>
<td>Annual maximum election amount that can be processed.</td>
</tr>
<tr>
<td>BENEFIT_REF_NUMBER</td>
<td>Employee’s reference number with the provider of the pension (benefit organization).</td>
</tr>
<tr>
<td>XML Tag</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
</tbody>
</table>

Managing HCM Extracts

Extract Components

The HCM Extracts feature is a flexible tool for generating data files and reports. This topic covers how you can use the extract components to define what information you want the application to extract and report on. It also explains how the application displays, formats, and delivers the information.

Extract Definitions

An extract definition refers to the complete setup of an extract, that consists of extract data groups, criteria, records, attributes, advanced conditions and output delivery options. An extract definition consists of:

- One or more extract data groups, depending on how many logical entities you want to extract.
- One or more extract records depending on how many groups of information you want to collect.
- One or more attributes depending on how many individual fields of data you want to collect.

You use HCM extracts to extract, archive, transform, report, and deliver high volumes of HCM data from the Oracle Fusion HCM database. You can generate the output in the following formats:

- CSV
- XML
- Excel
- HTML
- RTF
- PDF

You can distribute the extracted information by email, fax and other delivery modes. Some common examples of extracts are: PDF payslips delivered to employees’ mailboxes, payroll or benefits data transferred to third-party service providers, HR and talent data exchange between Oracle Fusion and legacy applications, for example in a coexistence scenario.

Data Groups

Extract data groups represent a business area or logical entity, for example person, assignment, or benefits. The application uses this information to retrieve the database item groups. You define one data group as the primary or root data group and this data group is the starting point of the data extraction.

Extract data group connections capture the association details between the current data group and the parent data group. The data group connections form the hierarchical relationship among the data groups.

You can define a set of filtering conditions the application must perform on an extract data group using the extract data group criteria. You specify the criteria conditions using an expression or fast formula.

Extract Records

Extract records represent a grouping of related data or a physical collection of all fields required in the extract. For example, the Employee data group can have records such as Basic Details, Pay Details, Location Details, and Primary Contact. An
An extract record is a collection of attributes which you can organize in a required sequence. For example, if a data group has 3 records, then you can specify the sequence in which the application processes the records. You can also select the next data group to identify which data group the application processes next.

**Attributes**

Attributes are the individual fields inside the extract record. An attribute is the lowest attribute level of a HCM extract and represents a piece of information, for example, person first name, person last name or person date of birth.

This figure demonstrates the hierarchy of information within a data group definition.

**Related Topics**

- Overview of Generating Flexfield Database Items

**HCM Extract Types**

The type of extract you select determines the purpose of the extract. It also determines the parameters that are automatically generated. For example, if you select the Payroll Interface extract type, then the application creates a changes only parameter, as well as the other parameters. You can select the extract type on the Manage HCM Extract Definitions page.

This table lists the different extract types and why you select them.
### Extract Type table:

<table>
<thead>
<tr>
<th>Extract Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Profile</td>
<td>Use for complete employee and payroll data archives.</td>
</tr>
<tr>
<td>Payroll Interface</td>
<td>Use for providing data to third party payroll service providers.</td>
</tr>
<tr>
<td>Payments</td>
<td>Use for salary payment method archives. For example, Paid through cheque or bank transfer.</td>
</tr>
<tr>
<td>Benefit Carrier</td>
<td>Use for providing data to third party benefits service providers.</td>
</tr>
<tr>
<td>Archive Retrieval</td>
<td>Use for reports based on permanently archived data, for example, payslip.</td>
</tr>
<tr>
<td>EOY Archive</td>
<td>Use for end of year archives (HR, Benefits).</td>
</tr>
<tr>
<td>HR Archive</td>
<td>Use for all HR archives.</td>
</tr>
<tr>
<td>Payroll Archive</td>
<td>Use for all payroll or payslip archives.</td>
</tr>
<tr>
<td>Other Payroll Archive</td>
<td>Use for all payroll archives.</td>
</tr>
</tbody>
</table>

### Payroll Interface Extract Definitions: Overview

The extract definitions of the Payroll Interface extract type determine the data you send your third-party payroll provider. The following table lists the predefined Payroll Interface extract definitions that you can use or define to meet your business-specific extract requirements.

<table>
<thead>
<tr>
<th>Extract Definition</th>
<th>Purpose</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Payroll Interface</td>
<td>Derives payroll data either from element entry values or from balance results created by the Calculate Gross Earnings process.</td>
<td>eText and XML</td>
</tr>
<tr>
<td>Payroll Interface for NGA</td>
<td>Extracts general-purpose HR and payroll-related data to integrate with Payroll Exchange for third-party payroll processing by NGA Human Resources. This extract transforms the data into an HR-XML format compliant with NGA Human Resources standards. Use the predefined Run Payroll Interface Report for NGA process if you don’t require special modifications.</td>
<td>XML</td>
</tr>
<tr>
<td>US ADP PayForce Third-Party Ad-Hoc Extract</td>
<td>Derives payroll data from element entries, including common HR and payroll data for a date range or a payroll period. Output format is compliant with Automatic Data Processing (ADP) PayForce standards.</td>
<td>eText and XML</td>
</tr>
</tbody>
</table>
Managing HCM Extracts

### Extract Definition

<table>
<thead>
<tr>
<th>Extract Definition</th>
<th>Purpose</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>US ADP PayForce Third-Party Periodic Extract</td>
<td>Derives payroll data from balance results created by the Calculate Gross Earnings process for a payroll period. Output format is compliant with ADP PayForce standards.</td>
<td>eText and XML</td>
</tr>
</tbody>
</table>

---

**Related Topics**

- Options to Extract Payroll-Related Data
- Example to Extract Payroll Data for Third-Party Processing

---

**Best Practices for Viewing User Entity Details**

When creating an extract definition, you need user entity details to create data groups. You can access the View User Entity Details UI using the Data Exchange tasks pane or when you create a data group in the Design tab of the Manage Extract Definitions task.

Using the View User Entity Details page, you can:

- View the database items available in a user entity.
- Validate user entities for use with root data groups.
- Review the type of user entity, for example, single row, multiple rows, historic, or effective dated.
- Review the SQL query the user entity uses.
- Calculate the number of rows in a data set.
- Review the contexts used and set by the user entity.
- Verify the extracts that are using the user entity.

The following sections detail the information that you can derive from the View User Entity Details UI.

### Reviewing the User Entity Details

Search for a user entity in the View User Entity Details UI and click the User Entity Details tab.

If the value of Valid for Root Data Group is Yes, the user entity can be used as a root data group. You can also determine if a user entity is valid for use as a root data group, if the value of Multiple Rows is Yes and Context Required is No. For example, the PER_EXT_SEC_PERSON_UE user entity can be used as a root data group, as the value of Multiple Rows is Yes and Context Required is No.

If a user entity is not a multiple row user entity, you must not use it as the root data group. If a multiple row user entity sets the contexts required for a single row user entity, the database item groups from the single row user entity are generally included as derived database item groups to the multiple row user entity. For example, the PER_PER_PERSON_DETAILS_UE user entity is not a multiple row user entity.

If a user entity is historic, it retrieves data irrespective of the effective date. For example, the PER_EXT_ASSIGNMENT_BASIC_HISTORY_UE user entity retrieves the entire historic details of an assignment.

If a user entity is not historic, it retrieves data as of the effective date. For example, the PER_EXT_PAY_EMPLOYEES_UE user entity retrieves Person, Assignment, and Payroll details as of the effective date.
Reviewing the SQL query
In the View User Entity Details UI, search for a user entity and click the Query tab. This tab lists the SQL query that the user entity uses to extract data. You can review the table structure and aliases which is useful for creating advanced filter criteria. If the user entity is valid as a root data group, you can calculate the rows that the SQL query will return. However, the number of rows returned is an estimate and may not match the exact number of rows the extract will return. The actual extract output depends on the filter criteria, expressions, and fast formulas used in the extract.

Reviewing the User Entity Contexts
In the View User Entity Details UI, search for a user entity and click the Context tab. The Contexts Required section lists the contexts that you have to set to use the user entity in the extract. For example, the PER_PER_PHONES_UE user entity requires the PERSON_ID context; therefore, you must set the PERSON_ID context to use this user entity. Generally, the user entities that are not historic require the EFFECTIVE_DATE context. However, the EFFECTIVE_DATE context is set by default, so you do not have to set this context explicitly. The Contexts Set section lists the contexts that are set by the user entity. For example, the PER_EXT_SEC_PERSON_UE user entity sets the PERSON_ID, ORGANIZATION_ID, and ENTERPRISE_ID contexts that can be used by other user entities.

Verifying Extracts Using the User Entity
In the View User Entity Details UI, search for a user entity and click the Extracts Using User Entity tab. If another extract is using the user entity, you can review the extract design to understand how the user entity is being used. You can also copy the extract design and then modify it to suit your needs.

User Entities in HCM Extracts
A user entity is a logical entity associated with a data group defined using HCM extracts. This topic describes the frequently used user entities and the type of data you can extract by using those user entities. You select user entities in the application when you define a data group.

The following table lists the most frequently used user entities.

<table>
<thead>
<tr>
<th>User Entity Name and Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Person (PER_EXT_SEC_PERSON_UE)</td>
<td>Retrieves all persons across the enterprise and all person related attributes.</td>
</tr>
<tr>
<td>Worker Payroll (PER_EXT_PAY_EMPLOYEES_UE)</td>
<td>Retrieves all workers and their payrolls across the enterprise, all person, worker, payroll related attributes, and element entry data.</td>
</tr>
<tr>
<td>Extract Assignment Basic History (PER_EXT_ASSIGNMENT_BASIC_HISTORY_UE)</td>
<td>Retrieves assignment history as on the assignment effective start date.</td>
</tr>
<tr>
<td>Extract Assignment Basic Information (PER_EXT_SEC_ASSIGNMENT_BASIC_UUE)</td>
<td>Retrieves assignment data as on the effective date.</td>
</tr>
<tr>
<td>Assignments Range (PER_EXT_SEC_ASSIGNMENT_RANGE_UE)</td>
<td>Retrieves assignment history as on the effective date.</td>
</tr>
<tr>
<td>User Entity Name and Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Extract Current and Future Persons (PER_EXT_SEC_PERSON_NOW_FUTURE_UE)</td>
<td>Retrieves current and future person details. Specify advanced filter criteria to restrict person types.</td>
</tr>
<tr>
<td>Extract Current and Future Assignments (PER_EXT_SEC_ASSIGNMENT_NOW_FUTURE_UE)</td>
<td>Retrieves current and future assignments.</td>
</tr>
<tr>
<td>Reset Context (PER_EXT_RESET_UE)</td>
<td>Use the Reset Context user entity in two ways:</td>
</tr>
<tr>
<td>1. Reset the effective date contexts set dynamically by a parent data group using a runtime input parameter for the effective date. For example, the Extract Assignment Basic History user entity sets the assignment effective start date as effective date first and then retrieves the data, unlike other user entities which use the effective date only. If you want to use the Extract Assignment Basic History user entity to include the historical data but also want to change the effective date, then include the Reset Context user entity to reset this date using input parameters.</td>
<td></td>
</tr>
<tr>
<td>2. Set up the Reset Context user entity as the root user entity to be a container or header. For example, the following work structure elements: locations, positions, and legal employers are not related to each other. If you want to extract all of them in one single extract, then you can add them as child elements to the Reset Context user entity used as the root in the extract definition. Adding this user entity as the root to these elements enables you to retrieve this data separately because the elements are not linked to a hierarchy.</td>
<td></td>
</tr>
</tbody>
</table>

You can view more details about the user entities using the View User Entity Details task.

**Related Topics**
- Example of Generating Flexfield Database Items

**HCM Extracts Templates**

Use the predefined Work Structure and Worker extract templates to quickly set up extracts that generate work structure and worker details. The Work Structure extract generates details of business units, legal entities, departments, grades, jobs, locations, and positions. The Worker extract generates details of person names, phones, emails, national identifiers, address, legislative data, accruals, absences, work relation, work term, assignment, work measure, supervisor, and salary.

To use these extracts, copy and configure them to suit your needs. You must not run the extract template directly. These extracts are complex and detailed. They can extract a huge volume of data and can be time consuming as they are designed on historic user entities to return all the details irrespective of the effective date.

When using these extract templates, ensure the following to avoid unneeded and time consuming data retrieval:
- Define filter criteria. For example, if you want to extract data related to specific locations, include an expression or fast formula in the data group filter conditions.
- Remove unwanted data groups. For example, if you are not interested in positions, then you can remove the positions data group.
- Review all the details to ensure all the information is relevant to your needs.
Using Work Structure and Worker Extract Templates

To use the Worker or Work Structure extract templates:

1. Copy the Worker or Work Structure extract definition to create a new extract.
2. Update the copied extract.
3. Generate and compile fast formulas.
4. Validate the extract design.
5. Refine the parameters.

Define Extracts

This example demonstrates the steps required to create an extract definition using the Desktop interface. You can access the Desktop interface by clicking the Switch Layout button on the Manage HCM Extract Definitions page. Before you create an extract definition, you should understand the following details:

- Information that you want to extract
- Structure in which the data must be extracted
- How you want to deliver this data (including file format, delivery mechanism, and frequency information)

FAST bank is a global organization with subsidiaries all over the world. As part of an external business reporting requirement, FAST bank is required to extract the department and employee details (grouped by department) across the entire company. This information must be sent to a third party in an XML file and to HR Managers in a PDF file using email. The following table summarizes the key decisions in this scenario:

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many extracts do I create to produce this type of report?</td>
<td>You create one extract definition to define a headcount report.</td>
</tr>
<tr>
<td>How many data groups do I create?</td>
<td>There are 2 functional groups of information, therefore you create two data groups, one for department and one for employees.</td>
</tr>
<tr>
<td>How many records do I create?</td>
<td>You decide the number of records based on the subgroup of attributes within a data group. In this example, you create two records for the department data group:</td>
</tr>
<tr>
<td></td>
<td>• Department Details</td>
</tr>
<tr>
<td></td>
<td>• Department Summary</td>
</tr>
<tr>
<td></td>
<td>You create one record for the employees data group: Employee Details.</td>
</tr>
<tr>
<td>How many attributes do I create?</td>
<td>You decide the number of attributes based on the specific information required for that report. Create the following attributes for the Department Details record:</td>
</tr>
<tr>
<td></td>
<td>• Department Name</td>
</tr>
<tr>
<td></td>
<td>• Department Location</td>
</tr>
<tr>
<td></td>
<td>For the Department Summary record, create the following attributes:</td>
</tr>
<tr>
<td></td>
<td>• Record Code</td>
</tr>
</tbody>
</table>
Decisions to Consider | In This Example
---|---
|  | Report Date
|  | Employee Count

For the Employees Details record, create the following attributes:

- Full Name
- Gender
- Date of Birth
- Salary
- Bonus
- Tax Rate

Do I create any fast formulas?

You can use fast formulas at the following levels:

- Extract Criteria level to determine certain conditions.
- Extract Rule level to derive attribute values.
- Extract Advanced Condition level to specify complex conditions.
- Extract Record level to automatically generate formulas when you use the Generate Formula option.

Creating an Extract Definition

1. On the Manage HCM Extract Definitions page, click Create.
2. Use the Switch Layout button to open the extract in the Desktop interface.
3. Enter 01-Jan-2000 as the Session Effective Date.
   The session effective date is an effective start date that applies to all date-effective interactions in the current session.
4. Enter FAST Bank Extract as the name and select HR Archive as the type. The application automatically creates the tag name based on the extract name and uses this name to generate the XML output file.
5. Click Save. The application saves the extract definition and automatically generates the parameters based on the type of extract. The parameters control the output of an extract. In this example, the application creates the following parameters:
   - Effective Date
   - Legislative Data Group
   - Parameter Group
   - Report Category
   - Request ID
   - Start Date

Creating Extract Data Groups

1. Select the Extract Data Group link from the navigation tree to open the Extract Data Groups region.
2. Click on Create to define a new data group. A data group represents data that belongs to one or more logical data entities.
3. Complete the fields to create a data group, as shown in this table:
4. Select Save and Create Another to create a data group for Employees.
5. Complete the fields to create a data group, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Employees</td>
</tr>
<tr>
<td>User Entity</td>
<td>PER_EXT_SEC_ASSIGNMENT_UE</td>
</tr>
<tr>
<td>Root Data Group</td>
<td>No</td>
</tr>
</tbody>
</table>

Creating Extract Data Group Connections

1. Select Extract Data Group in the navigation tree to display the data groups in a table.
2. Select the Employees Data Group and define the data group connection details.
3. Complete the fields to create a data group connection, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Data Group</td>
<td>Departments</td>
</tr>
<tr>
<td>Parent Data Group Database Item</td>
<td>PER_EXT_ORG_ORGANIZATION_ID</td>
</tr>
<tr>
<td>Data Group Database Item</td>
<td>PER_EXT_ASG_ORG_ID</td>
</tr>
</tbody>
</table>

4. Define the data group criteria for each data group.

Creating Extract Records

1. Select the Departments Data Group from the navigation tree and select the Create icon in the Extract Records region. Extract records represent a physical collection of all required fields. If a data group has 3 records, then you can specify the sequence in which the application processes the records using the sequence field. You can also select the Next Data Group to identify which data group the application processes next.
2. Create two records for the Departments data group.
3. Complete the fields to create two records for the Departments data group, as shown in this table:
4. Save the records, then select the Employees Data Group and select the Create icon in the Extract Records region.
5. Create one record for the Employees Data Group.
6. Complete the fields to create a record for the Employees data group, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Employee Details</td>
</tr>
<tr>
<td>Effective Start Date</td>
<td>1/1/00</td>
</tr>
<tr>
<td>Sequence</td>
<td>10</td>
</tr>
<tr>
<td>Type</td>
<td>Detail Record</td>
</tr>
<tr>
<td>Process Type</td>
<td>Fast Formula</td>
</tr>
</tbody>
</table>

Creating Attributes

1. Select the Departments Data Group from the navigation tree and select the Department Details record. An extract attribute is an individual field of a record.
2. Create the following extract attributes for the Department Details record and select Save.
3. Complete the fields to create extract attributes for the Department Details record, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Department Name</td>
<td>Department Location</td>
</tr>
<tr>
<td>Type</td>
<td>Database item group</td>
<td>Database item group</td>
</tr>
<tr>
<td>Database Item Group</td>
<td>Organization Name</td>
<td>Organization Location Country</td>
</tr>
</tbody>
</table>
4. Save the record, then select the Department Summary record.
5. Select the Create icon in the Extract Attributes region.
6. Create the following extract attributes for the Department Summary record and select Save.
7. Complete the fields to create extract attributes for the Department Summary record, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Record Code</td>
<td>Report Date</td>
<td>Employee Count</td>
</tr>
<tr>
<td>Data Type</td>
<td>Text</td>
<td>Date</td>
<td>Number</td>
</tr>
<tr>
<td>Type</td>
<td>String</td>
<td>Parameter Element</td>
<td>Summary Element</td>
</tr>
<tr>
<td>String Value</td>
<td>999</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Parameter</td>
<td>Effective Date</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Aggregate Function</td>
<td>NA</td>
<td>NA</td>
<td>Count</td>
</tr>
<tr>
<td>Aggregate Record Name</td>
<td>NA</td>
<td>NA</td>
<td>Employees Employee Details</td>
</tr>
</tbody>
</table>

8. Select the Employees Data Group from the navigation tree and select the Employee Details record.
9. Create the following extract attributes for the Employee Details record and select Save.
10. Complete the fields to create extract attributes for the Employee Details record, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Full Name</td>
<td>Gender</td>
<td>Date of Birth</td>
</tr>
<tr>
<td>Start Date</td>
<td>1/1/00</td>
<td>1/1/00</td>
<td>1/1/00</td>
</tr>
<tr>
<td>Data Type</td>
<td>Text</td>
<td>Text</td>
<td>Date</td>
</tr>
<tr>
<td>Type</td>
<td>Database Item Group</td>
<td>Decoded database item group</td>
<td>Database item group</td>
</tr>
<tr>
<td>Database Item Group</td>
<td>Person Full Name</td>
<td>Person Gender</td>
<td>Person Date of Birth</td>
</tr>
</tbody>
</table>

11. Complete the fields to create more extract attributes for the Employee Details record, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Salary</td>
<td>Bonus</td>
<td>Tax rate</td>
</tr>
<tr>
<td>Start Date</td>
<td>1/1/00</td>
<td>1/1/00</td>
<td>1/1/00</td>
</tr>
</tbody>
</table>
Defining the Delivery Options

1. Navigate to the Extract Execution Tree to validate the extract definition setup.
2. Select Export XSD to download the XML Schema Definition (.XSD) file for this extract setup. This exported file contains the structure of the extract definition: the data groups, records, and attributes.
3. Select the Extract Delivery Options region to define the formatting and layout options for the extract definition.
4. Complete the fields for the delivery options, as shown in this table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date</td>
<td>1/1/00</td>
<td>1/1/00</td>
</tr>
<tr>
<td>End Date</td>
<td>12/31/12</td>
<td>12/31/12</td>
</tr>
<tr>
<td>BI Publisher Template</td>
<td>ReportLayout</td>
<td>EFTLayout</td>
</tr>
<tr>
<td>Output Type</td>
<td>PDF</td>
<td>EFT</td>
</tr>
<tr>
<td>Delivery Type</td>
<td>Email</td>
<td>FTP</td>
</tr>
<tr>
<td>Delivery Option Name</td>
<td>Email to HR</td>
<td>FTP to 3rd Party</td>
</tr>
<tr>
<td>Output Name</td>
<td>HeadcountReport</td>
<td>EFTReport</td>
</tr>
</tbody>
</table>

5. Define further information for each delivery option in the Additional Details region. For example, add the server, username and password for the FTP delivery type.
6. Enter FAST Bank Extract as the reporting category and click Submit.
Submitting an Extract

An extract definition automatically creates an extract process (payroll flow) with the same name as the extract. The extract process enables you to define an execution sequence of multiple tasks, including before and after tasks.

1. Select the Submit Extracts task and select the FAST Bank Extract process.
2. Select Next.
3. Enter FAST Bank Extract - Jan 2012 as the Payroll Flow (extract process).
4. Enter 1/1/15 as the End Date.
5. Select Next. You can specify interaction details if the task is dependent on other tasks with different extract processes. For example, this task must wait because another task is running.
6. Select Next and review the extract. You can schedule the extract, or run it immediately.
7. Select Submit.
8. Select OK and View Checklist to view the status of the process.
9. Select the View Extract Results task to review the results of the extract run. Search for the FAST Bank Extract process.
10. Select Go to Task for FAST Bank Extract - Jan 2012, click the eyeglasses, and view the report output by selecting the report name.

Related Topics
- Guidelines for Delivering Extracts
- How do I create a BI Publisher template for HCM extract

Define Extracts in the Simplified Interface

This example topic demonstrates how to create a HCM extract including creating data groups, records, and attributes using the Simplified interface. FAST Bank is a global organization with subsidiaries all over the world. As part of an external reporting requirement, FAST Bank must obtain the department and employee details across the entire company. This information must be sent to a third party in an XML file and to the HR manager with employee details grouped by department as a Headcount Report.

The following table summarizes the key decisions in this scenario:

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>How many extracts do I create to produce this type of report?</td>
<td>You create one extract definition to define a headcount report.</td>
</tr>
<tr>
<td>How many data groups do I create?</td>
<td>There are 2 functional groups of information, therefore you create two data groups, one for department and one for employees.</td>
</tr>
<tr>
<td>How many records do I create?</td>
<td>You decide the number of records based on the subgroup of attributes within a data group. In this example, you create two records for the department data group:</td>
</tr>
<tr>
<td></td>
<td>• Department Details</td>
</tr>
<tr>
<td></td>
<td>• Department Summary</td>
</tr>
</tbody>
</table>
Decisions to Consider | In This Example
--- | ---
You create one record for the employees data group: Employee Details. |  
How many attributes do I create? | You decide the number of attributes based on the specific information required for that report. Create the following attributes for the Department Details record:  
• Department Name  
• Department Location  
For the Department Summary record, create the following attributes:  
• Record Code  
• Report Date  
• Employee Count  
For the Employees Details record, create the following attributes:  
• Full Name  
• Gender  
• Date of Birth  
• Salary  
• Bonus  
• Tax Rate
Do I create any fast formulas? | You can use fast formulas at the following levels:  
• Extract Criteria level to determine certain conditions.  
• Extract Rule level to derive attribute values.  
• Extract Advanced Condition level to specify complex conditions.  
• Extract Record level to automatically generate formulas when you use the Generate Formula option.

Creating an Extract Definition

1. On the Manage Extract Definitions page, click Create.  
2. Enter FAST Bank Extract as the name, 01-JAN-2010 as the Start Date, and select HR Archive as the type. The application uses this name to generate the XML output file.  
3. Click Save. The application saves the extract definition and automatically generates the parameters based on the type of extract. The parameters control the output of an extract.  
   Use the Switch Layout button to open the extract in the Desktop interface. Use the Desktop interface to create and define HCM extracts without using a drag and drop system. You can perform most of the tasks for defining the extract in the Simplified interface.

Creating Extract Data Groups and Records

1. Select the Design icon to create the data groups and records.  
2. Select the Create icon or use the HCM Data Objects tree to drag and drop a data group into the local area.
3. Using the drag and drop action, drag the Organizations object into the local area. This action automatically creates an associated record object.
4. Update the data group with the following information:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Departments</td>
</tr>
<tr>
<td>User Entity</td>
<td>PER_EXT_SEC_ORGANIZATION_UE</td>
</tr>
<tr>
<td>Root Data Group</td>
<td>Yes (By selecting this option you select this data group as the starting point for the extract execution.)</td>
</tr>
</tbody>
</table>

5. Update the record object and create another using the following information.
6. Complete the fields to create two records for the Departments data group, as shown in this table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Department Summary</th>
<th>Department Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Department Summary</td>
<td>Department Details</td>
</tr>
<tr>
<td>Effective Start Date</td>
<td>1/1/00</td>
<td>1/1/00</td>
</tr>
<tr>
<td>Sequence</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>Type</td>
<td>Trailer Record</td>
<td>Header Record</td>
</tr>
<tr>
<td>Process Type</td>
<td>Fast Formula</td>
<td>Fast Formula</td>
</tr>
<tr>
<td>Next Data Group</td>
<td>NA</td>
<td>Employees</td>
</tr>
</tbody>
</table>

7. Select Save and create another data group by dragging the Employees data object to the local area. This action automatically creates an associated record object.
8. Update the data group with the following information:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Employees</td>
</tr>
<tr>
<td>User Entity</td>
<td>PER_EXT_SEC_ASSIGNMENT_UE</td>
</tr>
<tr>
<td>Root Data Group</td>
<td>No</td>
</tr>
</tbody>
</table>

9. Update the record object with the following information.
10. Complete the fields to create a record for the Employees data group, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Employee Details</td>
</tr>
</tbody>
</table>
Creating Extract Data Group Connections

1. Select the Link icon on the Employees data group, and select the Add icon to add a connection.
2. Complete the fields to create a data group connection, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Data Group</td>
<td>Departments</td>
</tr>
<tr>
<td>Parent Data Group Database Item</td>
<td>PER_EXT_ORG_ORGANIZATION_ID</td>
</tr>
<tr>
<td>Data Group Database Item</td>
<td>PER_EXT_ASG_ORG_ID</td>
</tr>
</tbody>
</table>

3. Define the data group filter criteria for each data group by selecting the Filter icon.

Creating Attributes

1. Select the Departments Details record in the Department data group, and within the Configure icon select the Create Attribute option.
2. Complete the fields to create extract attributes for the Department Details record, as shown in this table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Attribute Entry</th>
<th>Attribute Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Department Name</td>
<td>Department Location</td>
</tr>
<tr>
<td>Type</td>
<td>Database item group</td>
<td>Database item group</td>
</tr>
<tr>
<td>Database Item Group</td>
<td>Organization Name</td>
<td>Organization Location Country</td>
</tr>
</tbody>
</table>

3. Select the Department Summary record, and using the above method enter the following extract attribute details.
4. Complete the fields to create extract attributes for the Department Summary record, as shown in this table:
### Defining the Delivery Options

1. Select the Deliver icon and then the Add icon to define the delivery options.
2. Complete the fields for the delivery options, as shown in this table:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date</td>
<td>1/1/00</td>
<td>1/1/00</td>
</tr>
<tr>
<td>End Date</td>
<td>12/31/12</td>
<td>12/31/12</td>
</tr>
<tr>
<td>BI Publisher Template</td>
<td>ReportLayout</td>
<td>EFTLayout</td>
</tr>
</tbody>
</table>
### Submitting an Extract

An extract definition automatically creates an extract process (payroll flow) with the same name as the extract. The extract process enables you to define an execution sequence of multiple tasks, including before and after tasks.

1. Select the Submit Extracts task and select the FAST Bank Extract process.
2. Select Next.
3. Enter FAST Bank Extract - Jan 2012 as the Payroll Flow (extract process).
4. Enter 1/1/15 as the End Date.
5. Select Next. You can specify interaction details if the task is dependent on other tasks with different extract processes. For example, this task must wait because another task is running.
6. Select Next and review the extract. You can schedule the extract, or run it immediately.
7. Select Submit.
8. Select OK and View Checklist to view the status of the process.
9. Select the View Extract Results task to review the results of the extract run. Search for the FAST Bank Extract process.
10. Select Go to Task for FAST Bank Extract - Jan 2012, click the eyeglasses, and view the report output by selecting the report name.

**Related Topics**

- Guidelines for Delivering Extracts

### How Extracts are Validated

Use the Validate feature in the Manage Extract Definitions task to ensure the data you enter in the Design tab is valid and there are no issues when you submit. After creating an extract, go to the Validate tab and click Validate. If there are any issues in the extract design, validation messages are displayed.
Validating Extracts

When you validate an extract, the application performs validations on the extract design to ensure that:

- The root data group is defined.
- All non-root data groups are linked to the root data group directly or indirectly (for example, through another non-root data group that is linked to the root data group).
- If a sequence of data groups is defined, then the next data group is also defined for processing.
- All Fast Formulas used in the extract exist and are compiled or valid.
- There are no issues detected during BI Publisher validations.

If the validation is successful, the application marks the extract as valid. However, if the validation fails, the application marks the extract as invalid and submitting the extract results in errors.

Review Extract Run Details

Using the View Extract Run Details task, you can review extract run information and troubleshoot extract runs that are unsuccessful or have not produced the expected results. In the Search area, use the various filters available to search the extract or extract run. In the Search Results area, click the extract run that you want to analyze.

In the View Extract Run Details page, you can review the following information:

- Parameters
- Archive details
- Process details
- Changes Only Details
- Delivery options

Parameters

The Parameters tab displays the parameters set for the extract run. For example, Baseline Only, Changes Only, Process Start Date, Process End Date, Process Configuration Group, and so on. You can also export these parameters to an Excel Spreadsheet.

Archive Details

The Archive Details tab displays the number of records that were extracted by the extract run.

Process details

The Process Details tab lists the processes and their hierarchy. You can view the process status, start and finish time, and the elapsed time. You can also download the process log files, which can help you troubleshoot failed or time consuming extract runs.
Extract Delivery Options
The Delivery Options tab is visible only if you have set the delivery option in the Manage Extract Definitions task as WebCenter Content or Inbound Interface. If the extract run is submitted and completed successfully, you can also download the output file.

Changes Only Details
Use the Changes Only Details tab to review the change in an attribute from the last successful run. This tab is visible only when the extract is run in the Changes Only mode. You can verify if the Changes Only parameter was set in the Parameters tab. Select the attribute and enter the attribute value to check the change in the attribute from the last successful run.

FAQs for Managing Extracts

What's the Simplified interface in HCM Extracts?
The Simplified interface is an easy-to-use graphical interface for defining and designing HCM extracts. You can perform most of the tasks for defining the extract in the Simplified interface, however to enter an effective date, you must use the Desktop interface.

What's an extract data group connection?
Data group connections enable you to define the master-detail of parent-child relationship between the entities. For example, the Employees and Departments data groups are linked with Department ID.

What's the difference between the Always Display and the Mark as Changed attributes?
You can see the difference between the Always Display and the Mark as Changed attributes when you have a parent child relationship in the extract. If an attribute is set as Mark as Changed, and a change occurs on the same record, then the application includes this attribute in the output. If no attributes change on the record, but an attribute changes in another record, for example a parent or sibling record, then the application does not include the attribute in the output.
If an attribute is set as Always Display and a change occurs on the record or in the hierarchy below the record, then the application includes this attribute in the output. The application includes the attribute even if there are no changes in the attribute’s record. If there is a change beneath the record, then it’s included.

Related Topics
- Guidelines for Using Different Modes for HCM Extracts
How can I disable an extract?

You can disable an extract by using the Manage Extract Definitions task in the Data Exchange area. Search for and open the extract that you want to disable. Change the status of the extract to Inactive. After disabling an extract, you cannot submit the extract run again. The previous submitted runs of a disabled extract are stored and are available for review.

Why can't I submit an extract run?

You can't submit an extract run when:

- The status of the extract is inactive. An inactive extract indicates that it's disabled. To submit an extract run, the status must be Active. Inactive extracts are not available for submission in the Submit Extracts task.
- The extract is invalid. After creating an extract, if you validate it and there are some validation errors, the extract is marked as invalid. If you try to submit an invalid extract, the process fails and errors are displayed. Resolve the validation errors and validate the extract. Then submit the extract run again.

Why can't I find my inactive extracts?

You can't find the inactive extracts because by default, the search results display active extracts only. To view inactive extracts, use the Manage Extract Definitions task in the Data Exchange work area. In the Search page, select Inactive in the Status drop-down list and click Search.

If you want the search results to display both active and inactive extracts, select the blank option in the Status drop-down list.

Do I have to validate existing extracts?

No. It's not required to validate existing extracts. Extracts that are not validated are marked as Not Yet Validated. You can submit extracts that are not validated. However, if there are issues with the extract design, then errors may occur or the result data may be incorrect. To avoid any errors, ensure that you validate the extract before submitting.

How can I download log files for the extract run processes?

Use the View Extract Results task. Search for the extract and select an extract run. In the Details area, the Process Details tab lists the extract run processes. You can download the log file for any process using the Log column. You can review the log files to identify the records that did not retrieve the expected data and to identify time consuming processes.

Can I use a user entity as a root data group?

Search for the user entity in the View User Entity Details UI. If it is a multiple row user entity and does not require any context, you can use this user entity as the root data group.
You can access the View User Entity Details UI:

- From the Data Exchange tasks pane.
- When creating a data group in the Design tab of the Manage Extracts Definitions task.

**Can I run a changes only extract with successful runs only?**

Yes. Use the Include changes from last successful run option to exclude the extract runs that did not complete successfully when you run a changes only extract. Leave the option unchecked if you want the application to include both the archives from extract runs that completed successfully and those that completed unsuccessfully when you run a changes only extract.

**How can I create and run a baseline extract?**

Use the Baseline Only parameter in your extract to create a baseline for which you can run all subsequent changes only extracts against. This parameter can save time when running the extract because it doesn’t generate an XML file and it doesn’t deliver any output. You set the Display option to Yes or Mandatory for the Baseline Only parameter. It creates a full extract in less time and uses less storage in your application.

**Can I create an extract and delete the archives?**

Yes. When you create a full extract of data, use the Delete Archive parameter to give you the flexibility of deleting archive and XML files at a later date. Set the Display option to Yes or Mandatory for the Delete Archive parameter. The application generates the archive and XML files and delivers them to the destination, for example, WebCenter Content. The Delete Archive parameter provides you with options to delete the archives and XML files after the application delivers the extract data to its destination. After the application delivers the output, for example, to WebCenter Content, you have the option to discard the archive data and XML files, therefore reducing storage consumption by deleting these extract archives.

You also have the flexibility to choose whether you discard all the data an extract generates, for example, use the Delete archive information and generated XML option to remove all unnecessary data. Or you can choose the Delete archive information only option to remove the archive data and retain the generated XML data. You may want to keep the XML data for reporting purposes or for future references.

**How can I improve the performance of the application during integration testing or when bulk data loads run?**

Contact your support representative and request them to manually gather the database table statistics for your pod. You may experience slow HCM Extract operations during certain testing phases of your implementation cycle, especially when the application tests bulk data loads at the same time. During regular operations the collection of statistics is scheduled to run weekly. However, if the data volume is high due to bulk data loads, then you should request this process to run manually.
### Guidelines for Using Different Modes for HCM Extracts

You can create an extract to output data in different modes in the Manage Extract Definitions page. Use modes to extract data that has changed since the previous extract runs. For example, you can extract employee details whenever there is a change in the employee's location. The processing engine generates the current status of the data, compares it with the base-lined data of the previous runs, and identifies the new as well as any modifications. The output from such changes-only extract has the incremental data only.

You can manipulate the output in a changes-only extract to either exclude or include certain attributes, regardless of whether the attribute has changed or not. For example, by selecting an option to exclude an attribute from comparison, you can ensure that that attribute is not compared while identifying changes to include in the generated output of the next extract run. Therefore, even if the attribute has changed since the previous extract run, it will not be included in the output. You can also select an option to always include an attribute in the generated output of an extract run, even if the attribute has not changed.

### Extract Modes

To enable different modes in an extract, you include the CHANGES_ONLY parameter and set up threading details in the extract definition. While running the extract you can use the CHANGES_ONLY parameter to control the extract mode. For example, to find out if the job name has changed on a person's assignment, set up multi-threading database items at assignment level and include the CHANGES_ONLY parameter. When you run the extract in the ATTRIBUTE mode, the extract compares data from the multi-threading level and outputs the incremental changes. You can link the PER_EXT_CHANGES_ONLY lookup or the ORA_HRY_CHANGES_ONLY lookup to the CHANGES_ONLY parameter so that you can select a mode when you submit the extract.

The following table describes the different extract modes, their lookup values and descriptions.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Lookup Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>All attributes</td>
<td>Includes all data in the extract. A full extract is run which produces the full data output at that point of time. The archived data is utilized as a baseline.</td>
</tr>
<tr>
<td>Y</td>
<td>Changed attributes</td>
<td>Compares this extract run with the previous extract runs and by comparing against the baseline (to identify the incremental data), displays the data that has changed only.</td>
</tr>
<tr>
<td>ATTRIBUTE</td>
<td>Changed and marked attributes</td>
<td>Includes elements that have changed or marked as mandatory</td>
</tr>
<tr>
<td>ATTRIB_OLD</td>
<td>Changed and marked attributes with previous values</td>
<td>Displays elements that have changed or marked as mandatory plus their previous value</td>
</tr>
</tbody>
</table>
Oracle Human Capital Management Cloud
Integrating with HCM

Chapter 34
Filtering and Delivering HCM Extracts

<table>
<thead>
<tr>
<th>Mode</th>
<th>Lookup Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOCK_OLD</td>
<td>Changed, marked attributes, previous data</td>
<td>Displays the following data under threading data group:</td>
</tr>
<tr>
<td></td>
<td>under threading group</td>
<td>Changed data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data marked as mandatory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Previous values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Parent data group values</td>
</tr>
<tr>
<td>BLOCK</td>
<td>Changed and marked attributes under threading</td>
<td>Displays the following data under threading data group:</td>
</tr>
<tr>
<td></td>
<td>group</td>
<td>Changed data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data marked as mandatory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Parent data group values</td>
</tr>
</tbody>
</table>

Note: You must run the Payroll Interface with the Attrib_ Old mode whenever you use the US ADP PayForce Third-Party Periodic Extract.

Filter Extracts Using Advanced Filter Criteria

Use advanced filter criteria to filter and extract data using complex SQL queries. You can also filter data using expressions or fast formulas. When using expressions, the available operators may not be sufficient to build complex expressions. You can use fast formulas to build complex filter criteria. However, using advanced filter criteria can improve the performance of the extract run. To use advanced filter criteria, you must have good understanding of the table aliases and writing SQL constructs.

Extracting Employees on Leave

This example explains how to use advanced filter criteria to extract employees who are going on leave in the next seven days. Managers can use this information to find the list of employees who have approved leaves starting in the next seven days.

1. On the Manage HCM Extract Definitions page, create an extract definition and use the Switch Layout button to open the extract in the Professional interface.
2. Select the Data Group link from the navigation tree to open the Data Groups page, and click Create.
3. Select the user entity PER_EXT_SEC_PERSON_UE and click Advanced.
4. In the Query tab, review the user entity tables and aliases that you want to use, and click OK.
5. In the Data Group Filter Criteria area, click Add.
6. Click the Edit icon in the Filter Criteria column.
7. In the Edit Filter Criteria Condition window, create the following expression for retrieving only employees: Extract Person System Person Type = 'EMP'
8. Click Advanced. The application converts the expression in the basic mode to the following SQL construct:

\[(\text{pptum.system_person_type} = \text{`EMP'})\]

9. You can now append new SQL construct by using the EXISTS clause. Entering the following SQL construct will extract employees who are going on approved leave in the next seven days:

\[
\{(\text{pptum.system_person_type} = \text{`EMP'}) \text{ and EXISTS}\}
\]

\[
\text{select 1 from fusion.ANC_PER_ABS_ENTRIES abs where abs.person_id = pptum.person_id and abs.start_datetime between pay_report_utils.get_parameter_value_date(`EFFECTIVE_DATE') and (pay_report_utils.get_parameter_value_date(`EFFECTIVE_DATE') + 7) and abs.approval_status_cd = `APPROVED'}\}
\]

When writing an SQL query, if you want to retrieve any parameter value at run time, use the `pay_report_utils.get_parameter_value_date(`ESS PARAMETER')` function. The ESS Parameter is typically, unless explicitly modified, the capitalized parameter name where any space is replaced by an underscore. For example, the ESS Parameter of Effective Date is EFFECTIVE_DATE.

Note: You cannot use more than 2000 characters to write the SQL query.

10. To validate the SQL construct for any syntactical errors, click Validate.
11. Click OK.
12. To verify if the advanced SQL criteria is applied correctly, run the extract in GMZFT logging mode and check the logs.

HCM Extract Formula Types

You can launch the Manage Fast Formulas UI from the Navigator Menu Payroll Calculation work area Clicking on the task Manage Fast Formulas.

The following table lists the different types of formula you can use for extracts:

<table>
<thead>
<tr>
<th>Formula Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Criteria</td>
<td>Formulas of this type are listed in the Data Group Criteria Fast Formula list. Use this type to specify complex criteria and to indicate if record should be extracted or not.</td>
</tr>
<tr>
<td>Extract Rule</td>
<td>Formulas of this type are listed in the Rule Type Attributes Fast Formulas list. Use this type to calculate and derive the attribute value if based on complex logic or business rules.</td>
</tr>
<tr>
<td>Extract Advanced Condition</td>
<td>Formulas of this type are listed in the Record Attribute Fast Formulas list. Use this type to specify complex conditions to indicate if an action should be raised (for example, a log or warning).</td>
</tr>
<tr>
<td>Extract Record</td>
<td>Formulas of this type are generated automatically when using the Generate Formula option in Extract Records.</td>
</tr>
</tbody>
</table>
Guidelines for Delivering Extracts

You specify delivery options as part of the extract definition. Delivery option parameters specify the output format, the delivery method (email, FTP), and other parameters required for integration with BI Publisher layer.

How Delivery Options Work

HCM Extracts archives the extracted data into result tables and stores it as an XML output in the database. The application transforms the XML output into a formatted output such as HTML, PDF, EDT, or XLS. The formatted output is then delivered through email, fax, FTP, or print, depending on the delivery options you set in the extract definition. You can define delivery options for an extract using a BI Publisher template, with the following delivery file output types: PDF, XLS, XML, DOC, and the following delivery modes: FTP, email, and fax.

Using Delivery Types

The type of delivery you select determines the destination of the extract. Some delivery types require additional information. You can select Documents of Record as the delivery mode to store the output in the database and allow employees to view the output from document of records. An example of a document is a payslip. If the XML output is split and burst as separate files, then you can select the bursting node. For example, if you want all employees to receive an email with their payslip, then set the bursting node to Employee_ID. Select the WebCenter Content delivery type to create extracts with encrypted or non-encrypted data and transmit them to Oracle WebCenter Content. You can then transfer the data manually or using your own scripts to your own server. For more information, see Oracle Fusion File Transfer Automation and Data Security on My Oracle Support.

Define the BI Publisher Template in HCM Extracts

This example shows you how to set up a BI Publisher template and create a document in the required format, using the extracted data. You can find BI Publisher using the Reports and Analytics link from the navigator. When the Reports and Analytics page is accessible, you can browse through the catalog.

FAST bank must send an XML file on their employees and departments to a third party and the HR Manager with employee details grouped by departments as a Headcount Report.

The following table summarizes some key decisions:

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In this Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why do I have to set up BI Publisher?</td>
<td>You set up BI Publisher to format the extracted XML data into the required format and to deliver the report or business document to the appropriate destination.</td>
</tr>
<tr>
<td>How does BI Publisher connect to HCM extracts?</td>
<td>You install the Template Builder for Word plug-in.</td>
</tr>
<tr>
<td>Do I need a separate license?</td>
<td>No, BI Publisher is included with Core HR.</td>
</tr>
</tbody>
</table>
Tasks

1. Setting up the Data Model
2. Creating the BI Publisher Report Template
3. Uploading the Report

Setting up the Data Model

1. Select the Reports and Analytics option from the navigator.
2. Select the Browse Catalog icon.
3. Select the New menu option, and under the Published Reporting section, select Report.
4. Select the Use the existing data model option to create a report using the existing data model.
5. In the window that displays, select the data model and select Next.
6. Select the Report Editor option and select Finish to complete the data model setup.
7. Save the report as HR Data Report in the following folder: /Shared Folders/Custom/Human Capital Management/

Creating the BI Publisher Report Template

1. Install the Template Builder for Word plug-in from the BI Publisher Enterprise Home page.
   This plug-in provides sample documents, demos, templates and Template Viewer. Use these samples to help you understand the concepts and to create templates such as EFT and RTF to view the formatted XML output.
2. Create a BI Publisher template using the Export XSD option in the extract execution tree.
3. Open MS Word and locate the Add-Ins tabbed region to view the Oracle BI Publisher option.
4. Select the Load XML Schema option. After the XML schema has loaded, you can arrange the fields and alter the layout, if required.

Uploading the Report

1. Select the Upload option from the Upload or Generate Layout region and enter the following information:

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layout Name</td>
<td>RTF Layout</td>
</tr>
<tr>
<td>Template File</td>
<td>Select the file name of the RTF template saved on your local drive.</td>
</tr>
<tr>
<td>Type</td>
<td>RTF Template</td>
</tr>
<tr>
<td>Locale</td>
<td>English (United States)</td>
</tr>
</tbody>
</table>

2. Save the BI report in the following location: /Shared Folders/Custom/Human Capital Management/.
   This BI Publisher report template is now ready for you to select in the Delivery Options section when creating the extract definition.
Best Practices for Delivering Encrypted Data Using HCM Extracts

You can deliver encrypted files using HCM Extracts to Oracle WebCenter Content. You use the WebCenter Content delivery type in the Deliver page of the Manage Extract Definitions task to create an encrypted extract. HCM Extract transfers the encrypted file to Oracle WebCenter Content. You can then transfer the files manually, or write scripts to perform the transfer. By selecting the WebCenter Content delivery type you can enter additional details to ensure the extract is identifiable in Oracle WebCenter Content. For example, specify the integration name, select whether you want to encrypt the file using the encryption mode, and a file name for your reference. This additional setup ensures the document has the correct encryption mode and is stored in Oracle WebCenter Content with a specific file name. For more information, see Oracle Fusion File Transfer Automation and Data Security on My Oracle Support.

FAQs for Filtering and Delivering HCM Extracts

How can I restrict the records to be extracted?

You can use extract data group criteria to define a set of filtering conditions the application performs on an extract data group. For example, you can use database items in the fast formula to represent the town_or_city and primary_flag columns in the per_addresses table to restrict the data to people living in London only. The extract definition would then exclude people with a primary address of anywhere other than London. You can specify the criteria conditions using an expression or fast formula.

What's a threading database item and what is its connection to the extract data group?

A threading database item is required for implementing the Changes Only feature. The threading database item is a unique ID in the chosen user entity. Generally, for Pay Employee user entity and Assignment user entity it would be DBI with %ASSIGNMENT%ID. For Person user entity, it would be DBI with pattern %PERSON%ID. You can declare one threading database item at the root data group or any child data group level. For example, you declare the threading database item from the location where you need changes only.
Can I use a user entity as a root data group?

Search for the user entity in the View User Entity Details UI. If it is a multiple row user entity and does not require any context, you can use this user entity as the root data group.

You can access the View User Entity Details UI:

- From the Data Exchange tasks pane.
- When creating a data group in the Design tab of the Manage Extracts Definitions task.

Related Topics

- Best Practices for Viewing User Entity Details

What's a conditional action?

A conditional action identifies the action to perform, and optionally, a message based on the outcome of a conditional expression or a predefined fast formula. Conditional actions are applied on the extracted data similar to criteria conditions that are applied prior to extraction. Actions and messages are predefined in lookups, and you can add your own messages by creating new values for the lookup.

When a condition is satisfied, you can use this feature to perform certain predefined actions. For example, you can exclude employees that satisfy a condition, such as all employees from a predefined country. You can also configure this feature to raise a warning when an employee’s salary is blank or beyond a specific level.

What's an exclusion rule?

You can’t exclude or override a record that doesn’t suit your requirements with your own record by using an exclusion rule. The extract process doesn’t process excluded records based on the legislative data group.

What is extract data group filtering?

Extract data group filtering filters data extracted in the data group. You can specify to filter data group as an expression or as fast formula, and you can choose one or both. You can build criteria using the available database items, parameters and operators (conditional and logical). The filter criteria will be more efficient as it gets appended to the User Entity SQL at the time of execution. If you can’t specify the criteria as an expression, then you can place logic inside a fast formula and select here. Fast formulas return values of Y or N to indicate if you must extract the record or not. If both criteria and formula are specified, then both the conditions are applied.

What are extract data group criteria?

You can specify the filter conditions of what data you want to archive by using the data group criteria. You can specify the filter conditions as an expression or fast formula.
Can I use Oracle Fusion Transactional Business Intelligence (OTBI) with HCM Extracts?

Yes. You have two options:

HCM Extracts can extract the data and produce an output using CSV, XML, or PDF. OTBI can then accept a data source in Excel or XML format.

BI Publisher can also accept a data source in Excel or XML format. HCM Extracts has integration with BI Publisher.

What's BI Publisher and how does it work with HCM extracts?

BI Publisher is a set of tools you use to create highly-formatted reports based on data models. With BI Publisher, you can:

- Author, manage, and deliver documents
- Create interactive management reports
- Create highly-formatted, customer-facing documents
- Create government documents
- Create electronic funds transfer (EFT) documents

BI Publisher transforms the extracted data from the database and presents that data into a report.

How do I create a BI Publisher template for HCM extract?

You create a BI Publisher template using the Export XML Schema option in the extract execution tree and saving the file to your local computer. You can then load the downloaded XSD file to the BI Publisher word plug-in using the XML Schema option. If you require a report in a specific format, then you can create a template and save it by arranging the fields in the required format. Otherwise, you can create a default RTF template using the All Fields option.

How can I disable bursting?

To disable the bursting feature in the BIP report and deliver a consolidated output:

- Select None as the delivery type on the Deliver page when creating a delivery option for the HCM extract.
- Select Disable in the Bursting field to disable the bursting options associated with the BIP template.

How can I reduce the size of my extract output?

In the Delivery tab, set the Compress option as Yes. You can use the compress feature with the WebCenter Content delivery option only. This feature extracts a compressed output, which helps to speed up large file transfers, uploading, and downloading. Compressing the extract output uses less disk space too.
Does a delivery option have to complete for an extract to run successfully?

No. Use the Required option in the Extract Delivery Option table to specify which delivery options you need to run successfully in order for the extract run to succeed. For example, use the Required option for critical delivery options such as payroll bursting. If this delivery option fails, then the extract run will also fail because it is dependent on the delivery option completing successfully.

You can also use the Required option for any delivery options you deem as not critical for the extract to run successfully. For example, deselect the Required option for a delivery option such as an email notification to yourself. If this delivery option fails, then the extract run will complete successfully because it is not dependent on the delivery option.

Why does my extract show a different time zone?

HCM Extracts uses the effective date to define the session date for a given extract run. The application uses this effective date to compare the effective start and effective end date attributes to filter the data it needs to process. The effective date (the session date) is the date set with the user time zone preferences. For example, if you run an extract with an effective date of 02-Jan-2018, with user time zone preference set to BST time zone, then the application retrieves the records created with an effective start date of 02-Jan-2018. Even if you create the data or perform a transaction on 01-Jan-2018 (or other dates), the effective start date here determines whether or not the transaction qualifies for the extract processing. The database stores the date with the time stamp in the UTC time zone for the attributes of the standard WHO columns. The date effective attributes (effective start date and effective end date used by the date effective feature in HCM Cloud) store the time without any time zone information.

Related Topics

- Can I add or edit time zones
- How can I set general preferences for all users
Flow Patterns

Flow Patterns for Extracts and Reports

A flow pattern exists for each extract or report process. Flow patterns for extracts and reports typically contain a single flow task and they define the submission parameters. Use the Manage Payroll Flow Pattern task from the Payroll Checklist work area or the Refine Extract task in the Data Exchange work area to edit flow patterns. You can edit a flow pattern to add additional tasks. Each flow pattern you create must be associated with a security profile.

Checklist and Flow Tasks

When you submit a flow pattern it creates an instance of the flow. For every submitted flow, the application generates a checklist by default. The flow can be a task flow, process, or report.

Here's what the checklist might include, depending on the flow pattern.

- Automatic tasks, such as extracts, reports, and processes
- Manual tasks, such as the Verify Payroll Process verification task, required to complete a flow

Here's what you can do with checklists.

- Monitor the status of the flow tasks
- Manage the flow tasks, such as reassigning tasks, marking tasks as completed, and performing corrective actions
- View task details, such as a list of records processed by the flow

While working on a task in the flow, you can remain in the Payroll Checklist work area or go to a related work area that includes tasks in the regional area. For example, while reviewing the results for the Calculate Payroll task, you can go to the Payroll Calculation work area to review the person's calculation card or element entries.

How Flow Patterns are Edited

You can create or copy a flow pattern and then edit add, delete, or move a task in the flow pattern. This topic provides examples of edits you can perform to tasks in a flow pattern or checklist.

Perform these edits on the Task Sequence tab of the Manage Payroll Flow Patterns page. For payroll flow patterns, use the Manage Payroll Flow Patterns task in the Payroll Checklist work area. For extract flow patterns, use the Refine Extracts task in the Data Exchange work area.

Edit Tasks

Here's a list of examples of edits you can perform and the probable impact the edits can have on the flow.
Edits | Impact | Examples
--- | --- | ---
Add a task | You add a task to position it as the last task in the activity or task group. Update the task sequence. If you repeat a task, rename it to make clear its purpose on the checklist. | You add a manual verification task after each report. You rename each task with the report name.

Delete a task | When you delete a task you may impact subsequent tasks in the flow that depend on its results. Review the subsequent tasks. | You delete a task. The Parameter Basis of the next task is Bind to Task and its Basis Value is the value of the deleted task. You update the Parameter Basis of the subsequent task as required, for example, to Bind to Flow.

Move a task to a different activity | The activity determines the work areas where you can submit the flow patterns you define, and controls how the checklist displays. | You move a task in a payroll flow pattern for a report from the Payments activity to the Statutory activity. The flow owner can view the report results from the Payroll Checklist or Regulatory and Tax Reporting work areas, but not the Payment Distribution work area.

Reorder the list of tasks displayed in a checklist | The sequence specified for the task further determines the task order within an activity on the checklist. | You decide to flatten the checklist sequence to group all the tasks within a single activity.

1. On the Tasks page, you confirm that each task belongs to the same activity and task group.
2. You edit each task, specifying a value in the Sequence column on the Edit Task Details Owners and Details page. The lowest number is used for the first task in the checklist. For example, you might specify a sequence of 10 for the payroll calculation task and 20 for the prepayments calculation task.

Flow Pattern Parameters

Each task in a flow pattern supports task actions, such as submit, roll back, mark for retry, retry, and view. Task action parameters control how the application processes a task and how the task relates to other tasks in the flow pattern.

Flow parameters are a subset of task action parameters. They supply the information required to successfully complete the tasks in the flow pattern.

- When you create a flow pattern, review and edit the task parameters for the Submit and Initialize task actions.
- Before you submit a flow, review and edit the task action parameters and the flow parameters for each task within the flow. Task action parameters control task interactions.
- After you submit the flow pattern, edit the parameters for the remaining task actions, such as Mark for Retry, Retry, and Roll Back, as required.
The following figure shows the relationship of the tasks, task action parameters, and flow parameters in a flow pattern.

Here’s the parameter details you can edit:

- Display and display format
- Lookups and value sets
- Usage
- Sequence
- Parameter Basis and Basis Value

**Display and Display Formats**

Display parameters control the format and availability of the flow parameter, as shown below.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display</td>
<td>Determines whether the parameter displays on the page when you submit the flow.</td>
</tr>
<tr>
<td>Display Format</td>
<td>Identifies the type of data displayed, such as a date or text, or choice list</td>
</tr>
</tbody>
</table>
Display parameters work with other parameters, such as Parameter Basis and Basis Value. For example, most task action parameters don’t display the Request parameter because the application obtains the value for this parameter from the context.

**Lookups and Value Sets**

Use lookups and value sets for descriptive flexfields to control and validate the data used in the payroll flow pattern.

Here’s the list of methods by which the lookup values are derived and the corresponding parameter basis you can use.

<table>
<thead>
<tr>
<th>Lookup Value</th>
<th>Parameter Basis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entered when submitting a flow</td>
<td>Bind to Flow</td>
</tr>
<tr>
<td>The application derives the value during flow submission and presents it. The value is derived from existing tables, such as the value for the payroll statutory unit. During flow submission, you can either use the value presented to you or override it.</td>
<td>SQL Bind or Bind to Flow Task or Context Binding</td>
</tr>
<tr>
<td>The application derives the value from a Post SQL process</td>
<td>Post SQL Bind</td>
</tr>
</tbody>
</table>

**Usage**

A parameter can receive information or generate information that subsequent tasks can use. For example, for the Calculate Payroll task, the Payroll Process parameter for the Submit task action generates an output value for the payroll action ID. The Retry task action can use this payroll action ID.

Here’s the typical settings for a parameter whose usage is output. For output usage parameters the parameter isn't displayed and its value is derived using the parameter basis.

<table>
<thead>
<tr>
<th>Parameter Option</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display</td>
<td>No</td>
</tr>
</tbody>
</table>
| Parameter Basis | • Bind to Flow, the application derives the value from the flow parameter and then updates the flow parameters table with the output value  
• If you select no value, the output value results from the task’s output |

**Sequence**

Sequence numbers control the order in which the application processes and displays the parameters by specifying the sequence. For example, if you have two lookups and the values of the second lookup depends on the first lookup. You must set the first lookup to a lower sequence number than the second one.
### Parameter Basis and Basis Value

The parameter basis controls how the application derives the value for the parameter. The basis value further specifies the value the application uses for the parameter.

Here’s the list of values to select parameter basis and basis values when you define payroll flows. The table provides examples when you can select them and describes how the values are assigned.

<table>
<thead>
<tr>
<th>Parameter Basis</th>
<th>Description</th>
<th>Basis Value Available</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Specified Value</td>
<td>Assigns a specific value to the parameter.</td>
<td>Enter the text as a constant or value, when you submit the flow.</td>
<td>Specify a constant if the value is the same for all tasks, such as the payroll statutory unit.</td>
</tr>
<tr>
<td>Bind to Context</td>
<td>Derives the value from the context of the current flow instance or the task instance of the flow pattern.</td>
<td>Select flow, task, or the Request. The application automatically generates the parameter value.</td>
<td>If the task includes a Request parameter, bind it to the flow context. Tasks in the flow reference this task using the Request ID generated by the application. Bind the legislative data group parameter to a task parameter that supplies the legislative data group. For example, the legislative data group for prepayments uses the payroll as context, because it’s already associated with the legislative data group.</td>
</tr>
<tr>
<td>Bind to Flow Parameter</td>
<td>Derives the value from one of the flow parameter values.</td>
<td>Application automatically derives the parameter value.</td>
<td>Bind a parameter to the flow that several tasks share to avoid multiple occurrences of the same parameter.</td>
</tr>
<tr>
<td>Bind to Flow Task Parameter</td>
<td>Binds the value to the output of the previous task.</td>
<td>Select a value from the previous task’s parameters.</td>
<td>Bind a parameter to a task, such as Retry corrective action. When the flow owner resubmits the task to retry it, the application uses the output of the Submit task parameter.</td>
</tr>
<tr>
<td>Bind to Task Parameter</td>
<td>Resolves the value for the task parameter.</td>
<td>Select a value from the current task’s parameters.</td>
<td>Bind a parameter to the task if several tasks share a parameter, such as a start date, but one task requires a different date.</td>
</tr>
<tr>
<td>No value specified</td>
<td>Stops the application from generating a parameter value when the task executes.</td>
<td>Application generates a blank value.</td>
<td>Not applicable</td>
</tr>
<tr>
<td>Post SQL Bind</td>
<td>Calculates the parameter but doesn’t display it on the user interface.</td>
<td>SQL statement calculates the parameter value.</td>
<td>Bind a parameter using the Post SQL bind to generate data.</td>
</tr>
</tbody>
</table>
### Flow Task Start and Due Dates

Use duration dates and notification options in the payroll flow pattern to give flow owners adequate time before a task starts, to prepare and address any issues.

#### Task Start and Due Dates

Let's look at the duration dates you can enter on the Tasks page of the Manage Payroll Flow Patterns page:

- **Start date**, the date the task owner should start the task.

  - **Note**: The start date applies to notifications only. Enter the flow start date on the Scheduling page when you schedule the flow.

- **Due date**, the date the task owner should complete the task.

To specify duration dates:

1. Select the flow parameter date to use as the basis for the duration date.
2. Optionally, offset the date by specifying a plus or minus value depending on whether the date falls before or after the duration date.

#### Notifications

Notifications are error and warning messages to inform the task owner when a task starts or ends. The receipt of notifications depends on the duration dates and their offsets.

1. Specify the notifications the task owner receives.
2. Optionally, specify the number of days before the application automatically deletes a notification from storage.

Use the Manage Payroll Process Configuration task to complete the Notification Expiration Offset parameter.
Examples of Editing Flow Patterns

Review these scenarios to understand how you can edit flow patterns to meet the requirements of your enterprise. Use the Manage Payroll Flow Pattern task in the Payroll Checklist work area or the Refine Extracts task in the Data Exchange work area to edit these scenarios.

Update a Parameter to Use a Specified Value

Your payrolls use a single process configuration group named InFusion Consolidation Group A. You want to specify a constant for the configuration group task action parameter and hide the parameter to avoid data entry mistakes. Perform these steps.

1. Query the flow pattern you defined for the payroll cycle.
   Enter these values to maintain a constant value for the Process Configuration Group task action parameter and avoid data entry mistakes.

<table>
<thead>
<tr>
<th>Parameter Detail</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display</td>
<td>No</td>
</tr>
<tr>
<td>Display Format</td>
<td>Text</td>
</tr>
<tr>
<td>Lookup</td>
<td>No value</td>
</tr>
<tr>
<td>Usage</td>
<td>Input Parameter</td>
</tr>
<tr>
<td>Parameter Basis</td>
<td>Constant Bind</td>
</tr>
<tr>
<td>Basis Value</td>
<td>InFusion Configuration Group A</td>
</tr>
</tbody>
</table>

Supply a Reason for a Corrective Action

Your enterprise typically issues electronic funds transfer payments. You defined a flow pattern to issue check payments and you want to verify and track the reason managers issue checks. You can add a flow parameter to capture that information.

1. Query the payments flow pattern you defined.
2. On the Parameters tab of the Manage Payroll Flow Pattern page, Select and Add the Reason parameter to include the parameter as a flow submission parameter. Enter these details.

<table>
<thead>
<tr>
<th>Parameter Detail</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Add Tasks and Reorder the Task Sequence

Your flow pattern includes the Calculate Gross Earnings process and the Element Results Register Report. Perform these steps to run the two extract reports concurrently, and add a verification task, to simplify the checklist to a single list.

1. From the Data Exchange work area, select the **Refine Extracts** task.
2. On the Refine HCM Extracts page, query the flow pattern.
3. On the Tasks tab of the Manage Payroll Flow Patterns page:
   a. Add the first extract report, specifying the same Activity and Task Group as the Calculate Gross Earnings.
   b. Add the second extract report, specifying the same Activity and Task Group as the Calculate Cross Earnings.
   c. Add a manual verification task, specifying the same Activity and Task Group as the Calculate Cross Earnings.
4. Edit each task, specifying a sequence number on the Edit Task Details Owners and Details page.

   The lowest number is used for the first task in the checklist. For example, you might specify a sequence of:
   - 10 for the Calculate Gross Earnings task
   - 20 for the first extract report
   - 30 for the second extract report
   - 40 for the manual verification task

5. On the Tasks Sequence tab, reorder the sequence of reporting tasks in this order to run the two reports concurrently.

<table>
<thead>
<tr>
<th>Task</th>
<th>Following Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Flow</td>
<td>Calculate Gross Earnings</td>
</tr>
<tr>
<td>Calculate Gross Earnings</td>
<td>First extract report</td>
</tr>
<tr>
<td>Calculate Gross Earnings</td>
<td>Second extract report</td>
</tr>
<tr>
<td>First extract report</td>
<td>Manual verification task</td>
</tr>
</tbody>
</table>
Automatically Increment Dates in the Scheduled Extract

You create a flow pattern to extract weekly payroll data that requires the user to enter a process date parameter. You schedule the extract to run weekly. The application evaluates the flow parameters at the time of submission, and the task parameters at the beginning of task execution. You edit the task parameters to automatically increment the date field. The date values are derived from the default date parameter values.

You use the Refine Extracts task from the Data Exchange work area, or the manage Flow Patterns task from the checklist work area. Perform these actions to edit the task parameters on the task’s Basic Information page.

1. Select the Process Date parameter.
2. Select Context binding from the Parameter Basis field.
3. Select System Date from the Basis Value field.

Add a BI Publisher Report to a Flow

Add single or multiple BI Publisher reports to your copied or user-defined flow pattern. When you submit the flow, the report automatically generates an output file that you can view. The output file is based on the template used for the BI Publisher report, such as an html template. The Run BI Publisher Report task belongs to the Statutory activities in the flow pattern.

Add Your Report to a Flow

Follow these steps to add the BI Publisher report to an existing extract flow.

1. Use the Manage Flow Patterns task in the Payroll Checklist work area or the Refine Extracts task in the Data Exchange work area.
2. Search and select the flow that you want to configure.
4. Click Select and Add on the menu bar. In the Search Tasks dialog, search for and select Run BI Publisher Report. Click Done.
5. On the Task Sequence page, confirm the sequence is correct.
6. On the Flow Parameters page, add a required parameter for the first argument of the BI Publisher report.

   The flow parameters map to the BI Publisher arguments. For example, if your report is based on a SQL query, the first argument is the first bind variable of a SQL query data model.

   Tip: To easily determine the sequence of arguments, view the list of parameters for the generated report in BI Publisher.

7. Optionally, rename the parameter to a more meaningful name.
8. On the Task parameters page, in the Parameter Details section, complete these steps:
   a. Confirm that the Parameter Basis for the First Argument value is Bind to Flow.
   b. Specify a value for the Report Name and Report Path parameters.
      For example, if the BI Publisher data model is saved to the Custom folder in Shared Folders you would specify
      /Custom/yourBIreport.xdo.
   c. Specify values for other arguments if required.
9. Review the flow and submit it.

Create a Flow within a Flow

In this example you copy the Transfer Batch flow and modify it to include a predefined report flow pattern you created earlier. The predefined flow you add submits a report to check for any batch line errors during the Transfer Batch process. If the transfer fails, you can skip the transfer process or mark it as complete, and then view the report for error details.

Create the Parent Flow Pattern

1. In the Payroll Checklist work area, select the Manage Payroll Flow Patterns task.
2. Search for and select the row for Transfer Batch, and then click Copy.
3. Enter the name of the new flow pattern, such as Transfer Batch with Error Report.
4. Enter a description, such as "Transfer a batch and view any batch line errors that occurred." and then click Save and Close.
5. Search for and select the Transfer Batch with Error Report flow pattern, and then click Edit.
6. To add the parameter that derives batch name from the batch ID:
   a. On the Parameters tab, click Add.
   b. Select the added row and click Edit.
   c. Add these values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow Parameter</td>
<td>Batch Name</td>
</tr>
<tr>
<td>Use for Searches</td>
<td>No</td>
</tr>
<tr>
<td>Display</td>
<td>No</td>
</tr>
<tr>
<td>Display Format</td>
<td>Text</td>
</tr>
<tr>
<td>Sequence</td>
<td>3</td>
</tr>
<tr>
<td>Usage</td>
<td>Input parameter</td>
</tr>
<tr>
<td>Parameter Basis</td>
<td>Post SQL Bind</td>
</tr>
<tr>
<td>Basis Value</td>
<td>select batch_name from pay_batch_headers where batch_id = :BATCH</td>
</tr>
</tbody>
</table>
7. Click **Save**.

**Add the Report Flow to the Parent Flow**

1. On the Tasks tab, click **Select and Add**.
2. In the Search window, search for and select **Submit Another Flow**, and then click **Done**.
3. In the row for **Submit Another Flow**, click **Edit** in the menu bar and set these values to define the task you have added to generate the batch lines error report.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>Run Batch Lines Report</td>
</tr>
<tr>
<td>Activity</td>
<td>Statutory</td>
</tr>
<tr>
<td>Task Group</td>
<td>Reporting</td>
</tr>
<tr>
<td>Description</td>
<td>Submit the batch lines error report for the specified batch.</td>
</tr>
</tbody>
</table>

4. Edit task parameters as follows:
   a. In the row for **Run Batch Lines Report**, click **Edit Task**.
   b. Configure these predefined task parameters. Select a parameter and the corresponding parameter basis and basis value.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Basis</th>
<th>Basis Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow Name</td>
<td>Constant Bind</td>
<td>The name of the flow, for example Batch Lines Report. This value is case-sensitive. Enter the name exactly.</td>
</tr>
<tr>
<td>From Flow Instance ID</td>
<td>Context Binding</td>
<td>Payroll flow</td>
</tr>
<tr>
<td>From Flow Task Instance ID</td>
<td>Context Binding</td>
<td>Payroll task</td>
</tr>
<tr>
<td>Use to Calculate Results</td>
<td>Constant Bind</td>
<td>Y</td>
</tr>
<tr>
<td>Parameter Name 1</td>
<td>Constant Bind</td>
<td>Batch Name</td>
</tr>
<tr>
<td>Parameter Value 1</td>
<td>Bind To Flow</td>
<td>Batch Name</td>
</tr>
</tbody>
</table>

c. Click **Next**, and optionally complete the owner and checklist information.
d. Click **Next**, and optionally complete the duration and notification information.
e. Click **Submit**.
5. Edit the task sequence as follows:
   a. On the Task Sequence tab, edit these two rows to enter the flow tasks and sequence in the flow.

<table>
<thead>
<tr>
<th>Start Flow</th>
<th>Following Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transfer Batch</td>
<td>Run Batch Lines Report</td>
</tr>
<tr>
<td>Run Batch Lines Report</td>
<td>End Flow</td>
</tr>
</tbody>
</table>

   b. Click **Submit**.

**Test the Flow**

1. Create and save a test batch that should cause an error.
   Alternatively, you can search for an existing batch that was transferred with errors using this SQL query:
   ```sql
   select * from pay_batch_headers where batch_status = 'E';
   ```
2. On the Submit a Process or Report page, select a legislative data group.
3. Select the **Transfer Batch with Error Report** task, and then click **Next**.
4. Enter a unique name for the current flow instance.
5. Enter the name of the batch with errors that you saved or queried, and then click **Submit**.
6. Click **OK and View Checklist**, and then click the **Refresh** icon until the Transfer Batch task shows as in progress with error.
7. View the report in the flow as follows:
   a. Select the row with the Transfer Batch task, and then select **Skip Task** in the Actions menu.
   b. In the row for Run Batch Lines Report, click **Go to Task**.
   c. In the Processes and Results section, click the name of the report.
   d. In the row for Run BI Publisher Report, click **Go to Task**.
   e. On the Process and Reports tab, click **View Results**.
   f. Click the PDF file name to open the report.

**FAQs for Flow Patterns**

What happens if I don't enter a task owner in a flow pattern?

The person who submits the flow becomes the flow owner and the task owner. The person’s security privileges determine whether the person can submit the flow.

How can I rearrange tasks in a flow pattern?

Every flow pattern begins with a Start Flow task, which doesn’t belong to an Activity or Task Group, and concludes with an End Flow task. Edit the task sequence by selecting a different task in the Following Task column.
When you create a flow, use the Task Sequence page to rearrange the tasks. When you edit a flow, select the task and edit its sequence on the Create Flow Pattern: Basic Information page. When you submit a flow, processes in the flow use and build upon the results of the previous processes. To maintain data integrity, ensure the sequenced tasks follow a consecutive order.

What happens if I change a due date for a task in a flow?

It doesn’t affect the status or progress information for the flow displayed on the checklist.

You must complete a flow task before beginning the next task. Before updating a due date beyond the next task’s start date, consider whether you have adequate time to perform the next task.

Only flow owners and task owners can update the due date.

Why don't the duration dates in the flow pattern display?

The start and end dates and their offsets display after you complete the flow parameter dates. Enter the flow parameters on the Parameters page, and then return to the Tasks page to enter the duration dates.

If your flow pattern doesn't specify dates as flow parameters, the duration list of values is blank. Change the values for the Duration list by displaying the date parameters for tasks in your flow pattern.

How can I improve performance and troubleshoot flows?

Add parameters to a payroll process configuration group to optimize performance and troubleshoot your payroll processes. To process large volumes of records, use the Threads and Chunk Size parameters. To troubleshoot processes, add the Logging Category or Formula Execution Logging parameters to a configuration group and rerun the process using that configuration group. Using these parameters enables you to investigate formula code problems.

Related Topics

- Payroll Process Configuration Groups
- Payroll Process Configuration Parameters

How can I run tasks concurrently in a flow?

Use the Manage Payroll Flow Patterns task in the Payroll Checklist. Search for and open your flow pattern. Click the Task Sequence tab. Specify that each concurrent task follows the same previous task.

For example, you create a flow pattern with a payroll calculation task and two reports. The flow ends when both reports complete. You can use these actions to run the payroll calculation task and the two reports run concurrently.

1. For the first row, specify the Start Flow task with the Calculate Payroll as the following task.
2. For the second row, specify Calculate Payroll as the task, and the first report as the following task.
3. For the third row, specify Calculate Payroll as the task, and the second report as the following task.
4. For the fourth row, specify the first report as the task and End Flow as the following task.
5. For the last row, specify the second report as the task and End Flow as the following task.
36 Benefits Data Extract

Guidelines for Extracting Benefits Data Using HCM Extracts

You can extract worker enrollment results, designated dependents and beneficiaries, and corresponding rates and coverages. You create an extract definition using the Manage Extract Definitions task in the Data Exchange work area. Consider these aspects before you create an extract definition for benefits data:

- Extract type
- Data groups
- Extract record sequence

Extract Type
When you create an extract definition, ensure that you select Benefits Carrier from the Extract Type list.

Data Groups
Create a PER_EXT_SEC_PERSON_UE (Person) data group and select it as the root data group. The following benefits data groups are available:

- BEN_EXT_ENRT_RSLT_UE (Enrollment Results)
- BEN_EXT_ENRT_DPNT_UE (Dependents)
- BEN_EXT_ENRT_BNF_UE (Beneficiaries)
- BEN_EXT_ENRT_RTCVG_UE (Rates and Coverages)

Extract Record Sequence
After you create the data groups, ensure that you sequence the data groups to identify which data group the application processes next:

- Person
- Dependents
- Beneficiaries
- Enrollments
- Rates and coverages

Related Topics
- Define Extracts
Configure Benefits for Extracting Data

This topic covers how to set up benefit plan carriers and offerings so that you can then generate and transmit enrollment data extracts. You extract benefits enrollment information into a single XML file for each benefits carrier.

To configure the benefits data extract, you complete these tasks in the Plan Configuration work area:

1. Create the plan carrier and configure extract settings.
2. Add the benefits extract plan type name.
3. Add the benefits extract plan code.
4. Add the benefits extract option name.

Creating Plan Carrier

Set up carrier data and configure extract options in the Plan Configuration work area that apply whenever you run the extract for that carrier.

1. In the Tasks panel drawer, click Manage Plan Carriers to open the Manage Plan Carriers page. On the Mapping tab, you can view the mapping of lookup codes to the lookup value that you transmit to carriers. Edits to mapping values affect all plan carriers that use the lookup.
2. On the Search Results toolbar, click the Create button.
3. Enter the carrier information. In the Active field, select Active.
4. Enter the extract options.

The following table provides comments to help you with your extract field entries and selections.

<table>
<thead>
<tr>
<th>Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Type</td>
<td>Select whether to run a full extract or extract only the changes since the previous extract. Generally, you run a full extract after an enrollment period closes and enrollments are completed. You run subsequent extracts on a periodic or scheduled basis, in either full or changes only mode.</td>
</tr>
<tr>
<td>Output File Name</td>
<td>Obtain the value that you enter here from the extract file recipient.</td>
</tr>
<tr>
<td>Processing Frequency</td>
<td>A common practice is to schedule your extracts to run after your regular payroll runs. You might want to set the processing frequency accordingly.</td>
</tr>
<tr>
<td>Processing Type</td>
<td>By default, all extracts have the same format, regardless of which carrier receives the extract. You can contract with Oracle’s partner, BenefitsXML, to have a carrier’s extract data transformed and delivered to the carrier, according to its specifications. For more information about BenefitsXML, see <a href="http://www.benefitsxml.com">http://www.benefitsxml.com</a>. If you don’t use this partner, you can transform and deliver the extract data file directly to each of your plan carriers, according to their specifications.</td>
</tr>
<tr>
<td>Upload Custom Layout</td>
<td>This button is available if you select the Custom layout processing type. Specify the layout for the individual carrier. For details, see the Benefits Extract: User-Defined Layout topic</td>
</tr>
</tbody>
</table>
5. Enter the file transfer details, which you obtained from the extract file recipient. You can transmit extract files directly to the recipient. Alternatively, you can transmit extract files to the Oracle cloud, using its file transfer details, and have your recipient download its extract from there.

<table>
<thead>
<tr>
<th>Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>The name of the server to which you transmit the recipient’s extract files.</td>
</tr>
<tr>
<td>Remote Folder</td>
<td>Path on the host to the location where your transmission places the extract file.</td>
</tr>
<tr>
<td>User Name</td>
<td>Part of the sign-in credentials required for your transmission to access the host.</td>
</tr>
<tr>
<td>Password</td>
<td>Part of the sign-in credentials required for your transmission to access the host.</td>
</tr>
</tbody>
</table>

6. Click **Save and Close** to return to the Manage Plan Carriers page.

Adding Benefits Extract Plan Type Name

Complete these steps in the Plan Configuration work area to identify each type of plan included in the extract file for the recipient.

1. In the Tasks panel drawer, click **Manage Plan Types** to open the Manage Plan Types page.
2. Search for and click the plan type, for example, Medical.
3. On the Plan Type Definition section **Actions** menu, select **Update**.
4. Enter the valid benefits extract plan type name, for example, Health.
5. Click **Save and Close** to return to the Manage Plan Types page.

Adding Benefits Extract Plan Code

Complete these steps in the Plan Configuration work area to link the plan to the extract file recipient.

1. In the Tasks panel drawer, click **Manage Benefits Plan Details** to open the Manage Benefits Plan Details page.
2. Search for and click the plan that you want to link to the extract file recipient.
3. Click **Next** to open the Edit Plan Additional Configuration page.
4. On the Configuration Details section **Actions** menu, select **Update**.
5. Enter the benefits extract plan code, which the extract file recipient provided to you for this specific plan.
6. Click **Save and Close** to return to the Plans tab.

Adding Benefits Extract Option Name

Complete these steps in the Plan Configuration work area to identify each option included in the extract file for the recipient.

1. In the Tasks panel drawer, click **Manage Benefit Options** to open the Manage Benefit Options page.
2. Search for and click the participant option, for example, Participant Only.
3. On the Basic Details section **Actions** menu, select **Update**.
4. Enter the valid benefits extract option name, for example, Employee Only.
5. Click **Save and Close** to return to the Manage Benefit Options page.
Next Steps

After you complete the previous tasks, you are ready to generate and transmit the extract data. The details of this process are covered in the Generating and Transmitting Benefits Data Extract for Plan Carriers: Procedure topic.

Valid Extract Names

Oracle partner BenefiX provides valid extract names that you can enter for benefits plan types and options. You can add values to this list, as required.

Benefit Extract Plan Type Names

Valid benefits extract names for plan types:

- 24 Care
- Dental
- Dental Capitation
- Exclusive Provider Organization
- Health
- Health Maintenance Organization
- Hearing
- Long Term Care
- Long Term Disability
- Mail Order Drug
- Major Medical
- Medicare Risk
- Mental Health
- Point of Service
- Preferred Provider Organization
- Prescription Drug
- Preventative Care
- Short Term Disability
- Utilization Review
- Vision

Benefits Extract Option Names

Valid benefits extract names for options

- Children Only
- Dependents Only
Generate and Transmit Benefit Extracts

You can extract benefits enrollment information into a single XML file for each benefits carrier and transmit it to the carrier. This topic covers how to generate and transmit the extract file.

**Note:** If a carrier provides more than one plan, the single extract contains information related to all plans provided by that carrier. Example: Four different carriers provide ten plans. You run four separate extracts, one for each carrier.

To generate and view the extract, you complete these tasks in the Evaluation and Reporting work area.

1. Submit the extract request.
2. View and transmit the extract details.

Prerequisites

You must first configure your plan carriers, or extract recipients, and add the relevant extract data to the appropriate plan types, plans, and options. The details of this process are covered in the Configuring Benefits Data Extract for Plan Carriers: Procedure topic.
Submitting Extract Request

To run and monitor extracts, in the Evaluation and Reporting work area:

1. In the Tasks panel drawer, click **Extract Benefits Data** to open the Extract Benefits Data page.
2. On the Search Results toolbar, click **Submit**.
3. Enter the extract request options.

<table>
<thead>
<tr>
<th>Field</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract Type</td>
<td>For a particular carrier, you can select whether to run a full extract or extract only the changes since you ran the previous extract.</td>
</tr>
<tr>
<td></td>
<td>- Generally, you run a full extract after an enrollment period closes and enrollments are completed.</td>
</tr>
<tr>
<td></td>
<td>- You run subsequent extracts on a periodic or scheduled basis, in either full or changes only mode. Common practice is to schedule your extracts to run after your regular payroll runs.</td>
</tr>
<tr>
<td>Transmit</td>
<td>You can transmit the extract as part of the extract request, or after the requested extract completes and before or after you view the extract details.</td>
</tr>
</tbody>
</table>

4. Click **Submit** to submit your process and return to the Extract Benefits Data page.

Viewing and Transmitting Extract Details

You can view, query, and download extracted records for a specific extract run after it completes. You can also transmit the extract after the requested extract completes and before or after you view the extract details.

1. In the Search Results section of the Extract Benefits Data page, click the **Request ID** for the most recent extract request for the plan carrier.
2. Review, query by example, and download to a spreadsheet the extracted data, as appropriate.
3. Click **Done** to return to the Extract Benefits Data page.
4. In the Search Results section, click the **Transmit** icon button for the most recent extract request.

Example of a Custom Layout for Benefits Extract

An implementor or developer can create a custom layout to transform the format of extracted benefits enrollment data to match the specifications of a particular carrier. This topic provides:

- Descriptions of the tags that you require to create the custom layout
- Table aliases and a sample custom layout

The custom layout becomes the default layout for the plan carrier after you upload it to the plan carrier’s extract options.

The following code shows the structure of the XML tags in the custom layout.

```xml
<Layout
  <Table
    Record Type
    <Field
      Name
      Source
```

ORACLE
The data source for a field on the custom layout can be a:

- Column on the benefits extract staging tables
- Column on one of the other tables listed in the Source tag description
- Constant into which you enter the exact value

Tip: To identify table column names, you can use the data model query builder in Oracle BI Publisher. Search for the table name and view the columns.

XML Tag Descriptions

This section describes each XML tag and lists its attributes, elements (subtags), and parent tags.

Layout

Description: Root tag.

Attributes: None

<table>
<thead>
<tr>
<th>Elements (Subtags)</th>
<th>Parent Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>None</td>
</tr>
</tbody>
</table>

Table

Description: Specifies the database table from which to extract the data.
Attributes: 1. tableName: Supported values = {BEN_EXTRACT_REQ_DETAILS, BEN_EXTRACT_REQUEST, DUAL, PER_ALL_PEOPLE_F, PER_PERSONS, PER_ALL_ASSIGNMENTS_M, PER_PEOPLE_LEGISLATIVE_F, BEN_PL_F, BEN_PL_TYP_F, ...

<table>
<thead>
<tr>
<th>Elements (Subtags)</th>
<th>Parent Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. RecordType</td>
<td>Layout</td>
</tr>
<tr>
<td>2. Field: See Field tag description</td>
<td></td>
</tr>
</tbody>
</table>

Record Type

Description: Specifies how to delimit or lay out the data in the extract file.

Attributes: Supported values: FIXEDWIDTH, CSV

> **Note:** Anything other than CSV is delimited as FIXEDWIDTH by default.

<table>
<thead>
<tr>
<th>Elements (Subtags)</th>
<th>Parent Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Table</td>
</tr>
</tbody>
</table>

Field

Description: Corresponds to one column in the extracted document. Source the text in this column from a database table, an SQL function, or a constant.

Attributes: None

<table>
<thead>
<tr>
<th>Elements (Subtags)</th>
<th>Parent Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Name</td>
<td>Table</td>
</tr>
<tr>
<td>2. Source</td>
<td></td>
</tr>
<tr>
<td>3. Width</td>
<td></td>
</tr>
<tr>
<td>4. Padding</td>
<td></td>
</tr>
</tbody>
</table>

Name

Description: Name of the field

Attributes: None
Source

Description: Specifies the source of data for the current field.

- If the source is a table, the value passed is the column name.
- If multiple tables are involved, use a fully qualified column name.

The list of allowed tables includes the table aliases.

SQL functions in place of column names: Values in this tag are treated as column names if the type is set to TABLE. The column name is used directly while constructing a query, so an SQL function can be used on a column.

- Example 1
  
  ```xml
  <Source type="TABLE">GENDER_FLAG</Source>
  ```

- Example 2
  
  ```xml
  <Source type="TABLE">DECODE(GENDER_FLAG, 'F', 1, 2)</Source>
  ```

Attributes:

1. type:
   - Supported values = {TABLE, CONSTANT}
     - TABLE specifies that the data comes from a database table.
     - CONSTANT specifies that the data is given in the value column of this tag.

2. table: Use this tag only if the intended column isn’t from the table given in the tableName attribute of this Table tag. If this tag isn’t used, the column is searched for in the table given in tableName.
   - Supported values:
     - {BEN_EXTRACT_REQ_DETAILS
     - PER_ALL_PEOPLE_F
     - PER_PERSONS
     - PER_ALL_ASSIGNMENTS_M
     - PER_PEOPLE_LEGISLATIVE_F
     - BEN_PL_F
     - BEN_PL_TYP_F
     - BEN_OPT_F
     - BEN_PGM_F

Elements (Subtags) | Parent Tag
--- | ---
None | Field
Width

Description: Specifies the intended width of this field in the extract file. The number passed is the number of character spaces on the file.

Attributes: Supported values are positive integers.

<table>
<thead>
<tr>
<th>Elements (Subtag)</th>
<th>Parent Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Field</td>
</tr>
</tbody>
</table>

Padding

Description: Specifies the alignment of data in each column.

Attributes: Supported values: {LEFT, RIGHT}

<table>
<thead>
<tr>
<th>Elements (Subtag)</th>
<th>Parent Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Field</td>
</tr>
</tbody>
</table>

Table Aliases

<table>
<thead>
<tr>
<th>Allowed Table</th>
<th>Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEN_EXTRACT_REQ_DETAILS</td>
<td>REQ</td>
</tr>
<tr>
<td>PER_ALL_PEOPLE_F</td>
<td>PEO</td>
</tr>
<tr>
<td>PER_PERSONS</td>
<td>PER</td>
</tr>
<tr>
<td>PER_ALL.Assignments_M</td>
<td>ASG</td>
</tr>
<tr>
<td>PER_PEOPLE_LEGISLATIVE_F</td>
<td>LEG</td>
</tr>
<tr>
<td>BEN_PL_F</td>
<td>PLN</td>
</tr>
<tr>
<td>BEN_PL_TYP_F</td>
<td>TYP</td>
</tr>
<tr>
<td>BEN_OPT_F</td>
<td>OPT</td>
</tr>
</tbody>
</table>
Sample XML Layout

```xml
<?xml version="1.0" encoding="utf-8"?>
<Layout>
  <Table tableName="DUAL">
    <RecordType>FIXEDWIDTH</RecordType>
    <Field>
      <Name>"Record Type"</Name>
      <Source type="CONSTANT">001</Source>
      <Width>3</Width>
      <Padding>Left</Padding>
    </Field>
  </Table>
  <Table tableName="BEN_EXTRACT_REQ_DETAILS">
    <RecordType>CSV</RecordType>
    <Field>
      <Name>"Last Name"</Name>
      <Source type="TABLE">LAST_NAME</Source>
      <Width>25</Width>
      <Padding>Left</Padding>
    </Field>
    <Field>
      <Name>"First Name"</Name>
      <Source type="TABLE">FIRST_NAME</Source>
      <Width>50</Width>
      <Padding>Left</Padding>
    </Field>
    <Field>
      <Name>"Filler"</Name>
      <Source type="CONSTANT">XXXXXXXXXX</Source>
      <Width>10</Width>
      <Padding,None</Padding>
    </Field>
    <Field>
      <Name>"Plan Name"</Name>
      <Source type="TABLE">PLAN</Source>
      <Width>70</Width>
      <Padding>Left</Padding>
    </Field>
    <Field>
      <Name>"Coverage Start Date"</Name>
      <Source type="TABLE">COVERAGE_START_DATE</Source>
      <Width>15</Width>
      <Padding>Left</Padding>
    </Field>
    <Field>
      <Name>"SSN"</Name>
      <Source type="TABLE">NATIONAL_IDENTIFIER</Source>
      <Width>12</Width>
      <Padding>Left</Padding>
    </Field>
    <Field>
      <Name>"Gender"</Name>
      <Source type="TABLE">DECODE(GENDER_FLAG, 'F', 1, 2)</Source>
      <Width>1</Width>
    </Field>
  </Table>
</Layout>
```
<table>
<thead>
<tr>
<th>Field</th>
<th>Source</th>
<th>Width</th>
<th>Padding</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERSON_NUMBER</td>
<td>&lt;Source type=&quot;TABLE&quot; table=&quot;PER_ALL_PEOPLE_F&quot;&gt;PERSON_NUMBER&lt;/Source&gt;</td>
<td>30</td>
<td>Left</td>
<td>&quot;Person Number&quot;</td>
</tr>
<tr>
<td>COUNTRY_OF_BIRTH</td>
<td>&lt;Source type=&quot;TABLE&quot; table=&quot;PER_PERSONS&quot;&gt;COUNTRY_OF_BIRTH&lt;/Source&gt;</td>
<td>30</td>
<td>Left</td>
<td>&quot;Country of Birth&quot;</td>
</tr>
<tr>
<td>ASSIGNMENT_TYPE</td>
<td>&lt;Source type=&quot;TABLE&quot; table=&quot;per_all_assignments_m&quot;&gt;assignment_type&lt;/Source&gt;</td>
<td>30</td>
<td>Left</td>
<td>&quot;Assignment type&quot;</td>
</tr>
<tr>
<td>LEGISLATION_CODE</td>
<td>&lt;Source type=&quot;TABLE&quot; table=&quot;per_people_legislative_f&quot;&gt;LEG.LEGISLATION_CODE&lt;/Source&gt;</td>
<td>30</td>
<td>Left</td>
<td>&quot;Legislation code&quot;</td>
</tr>
<tr>
<td>PLN_ID</td>
<td>&lt;Source type=&quot;TABLE&quot; table=&quot;ben_pl_f&quot;&gt;PLN.PL_ID&lt;/Source&gt;</td>
<td>30</td>
<td>Left</td>
<td>&quot;Legislation code&quot;</td>
</tr>
<tr>
<td>RECORDCOUNT</td>
<td>&lt;Source type=&quot;CONSTANT&quot;&gt;999&lt;/Source&gt;</td>
<td>3</td>
<td>Left</td>
<td>&quot;Record Type&quot;</td>
</tr>
<tr>
<td>RECORDCOUNT</td>
<td>&lt;Source type=&quot;SYSTEM&quot;&gt;RECORDCOUNT&lt;/Source&gt;</td>
<td>3</td>
<td>Left</td>
<td>&quot;Record Type&quot;</td>
</tr>
</tbody>
</table>
37 Talent Review and Succession Management Data Extract

Guidelines for Extracting Talent Review Data Using HCM Extracts

You can use HCM Extracts to extract Talent Review data. Here’s the data that you can extract with a few examples of the attributes that you can extract for each:

- **Meeting details**: Meeting ID, meeting dashboard template ID, meeting date
- **Facilitators’ details**: Facilitator ID, facilitator person number, facilitator creation date
- **Participants’ details**: Participant ID, participant business group ID, participant person number
- **Talent pools’ details**: Talent pool ID, talent pool meeting ID, talent pool creation date
- **Prior rating details**: Prior rating date range ID, prior rating date range meeting ID, prior rating date range creation date
- **Reviewees’ details**: Reviewee ID, reviewee meeting ID, reviewee person number
- **Review content details**: Review content ID, review content meeting ID, review competencies
- **Succession plans’ details**: Succession Plan ID, succession plan meeting ID, succession plan creation date

In this topic, we discuss only the information that you need for extracting Talent Review data. For more information about creating an extract definition and publishing the extract results, see the related topics.

Create Talent Review Extract Definition

To create an extract definition, you must determine these details:

- The user entity details to create data groups
- The database item group details to create attributes
- The extract definition type

You create an extract definition using the Manage Extract Definitions task in the Data Exchange work area.

This table lists the Talent Review related user entities and the starting characters of the database items that correspond to these user entities. You can search the Talent Review related database items using these characters.

<table>
<thead>
<tr>
<th>Talent Review Data to Extract</th>
<th>User Entity</th>
<th>Database Items Start With</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meeting details</td>
<td>HRR_EXT_MEETINGS_</td>
<td>HRR_MEETINGS_</td>
</tr>
<tr>
<td>Meeting facilitators</td>
<td>HRR_EXT_MEETING_FACILITATORS</td>
<td>HRR_MEETING_FACILITATORS_</td>
</tr>
<tr>
<td>Meeting participants</td>
<td>HRR_EXT_MEETING_PARTICIPANTS</td>
<td>HRR_MEETING_PARTICIPANTS_</td>
</tr>
</tbody>
</table>
Talent Review Data to Extract | User Entity | Database Items Start With
--- | --- | ---
Associated talent pools | HRR_EXT_MEETING_TAENTPOOLS | HRR_MEETING_TAENTPOOLS_
Meeting prior rating date ranges | HRR_EXT_PRIOR_RATING_DATE_RANGES | HRR_PRIOR_RATING_DATE_RANGES_
Meeting review population | HRR_EXT_MEETING_REVIEWEES | HRR_MEETING_REVIEWEES_
Meeting review content | HRR_EXT_MEETING_REW_CONTENT | HRR_MEETING_REW_CONTENT_
Associated succession plans | HRR_EXT_MEETING_SUCCPLANS | HRR_MEETING_SUCCPLANS_

While creating the Talent Review extract definition, select the extract definition type as **HR Archive**.

With the information in this topic, you're now ready to extract Talent Review data.

**Related Topics**
- Extract Components
- HCM Extract Types
- Define Extracts
- Filter Extracts Using Advanced Filter Criteria
- Define the BI Publisher Template in HCM Extracts

**Guidelines for Extracting Succession Management Data Using HCM Extracts**

You can use HCM Extracts to extract Succession Management data.

Here's the Succession Management data that you can extract with a few examples of the attributes that you can extract for each:

- **Plan details**: Plan type, status, incumbent ID
- **Candidate details**: Candidate person number, status, readiness value
- **Owner details**: Owner person number, meaning of the owner type code

In this topic, we discuss only the information that you need for extracting Succession Management data. For more information about creating an extract definition and publishing the extract results, see the related topics.

**Create Succession Management Extract Definition**

To create an extract definition, you must determine these details:

- The user entity details to create data groups
• The database item group details to create attributes
• The extract definition type

You create an extract definition using the Manage Extract Definitions task in the Data Exchange work area.

This table lists the Succession Management related user entities and the starting characters of the database items that correspond to these user entities. You can search the related database items using these characters.

<table>
<thead>
<tr>
<th>Succession Management Data to Extract</th>
<th>User Entity</th>
<th>Database Items Start With</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan details</td>
<td>HRM_EXT_PLANS_UE</td>
<td>HRM_PLANS_</td>
</tr>
<tr>
<td>Plan candidates</td>
<td>HRM_EXT_PLAN_CANDIDATES_UE</td>
<td>HRM_PLAN_CANDIDATES_</td>
</tr>
<tr>
<td>Plan owners</td>
<td>HRM_EXT_PLAN_OWNERS_UE</td>
<td>HRM_PLAN_OWNERS_</td>
</tr>
</tbody>
</table>

While creating the Succession Management extract definition, select the extract definition type as **HR Archive**.

With the information in this topic, you’re now ready to extract Succession Management data.

**Related Topics**

• Extract Components
• HCM Extract Types
• Define Extracts
• Filter Extracts Using Advanced Filter Criteria
• Define the BI Publisher Template in HCM Extracts
Transferring Encrypted Data

Set up Encryption for File Transfer

You use encryption keys to encrypt files for secure transfer between Oracle HCM Cloud and your own servers through the Oracle WebCenter Content server. This PGP-based encryption support is available for secure file transfer using HCM Data Loader and HCM Extracts.

The process for inbound files (into Oracle HCM Cloud) is:

1. You encrypt files using the Oracle HCM Cloud public key.
2. The data-loading process decrypts files using the Oracle HCM Cloud private key.

The process for outbound files (generated by Oracle HCM Cloud) is:

1. Oracle HCM Cloud encrypts files using your public key.
2. You decrypt files using your private key.

Therefore, before you can encrypt or decrypt files, you must:

1. Import your public key to Oracle HCM Cloud.
2. Generate the PGP encryption key pair and download the Oracle HCM Cloud public key.

You can also sign the files, as follows:

• Outbound files are signed using the HCM Cloud private key. You verify these files using the HCM Cloud public key.
• Inbound files are signed using your private key. The data-loading process verifies inbound files using your public key.

This topic describes how to set up encryption, decryption, and signing of files.

Encryption and Signature Keys

This table shows you the keys that are used for encryption, decryption, signing, and verification in each supported encryption mode.

<table>
<thead>
<tr>
<th>Encryption Mode</th>
<th>Encryption Key</th>
<th>Decryption Key</th>
<th>Signing Key</th>
<th>Verification Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outbound PGP Signed</td>
<td>customer-key_pub</td>
<td>customer-key_priv</td>
<td>fusion-key_priv</td>
<td>fusion-key_pub</td>
</tr>
<tr>
<td>Outbound PGP Unsigned</td>
<td>customer-key_pub</td>
<td>customer-key_priv</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Inbound PGP Signed</td>
<td>fusion-key_pub</td>
<td>fusion-key_priv</td>
<td>customer-key_priv</td>
<td>customer-key_pub</td>
</tr>
<tr>
<td>Inbound PGP Unsigned</td>
<td>fusion-key_pub</td>
<td>fusion-key_priv</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Importing Your Public Key

You use your public key (customer-key_pub) for encrypting outbound files. You can decrypt the files using your private key (customer-key_priv). If you also want outbound files to be signed, then use the HCM Cloud private key (fusion-key_priv) for signing. You can verify signed outbound files using the HCM Cloud public key (fusion-key_pub).

To import the customer public key:

1. Sign in to Oracle HCM Cloud with the IT Security Manager job role or privileges.
3. Click the Certificates tab to open the Certificates page.
4. Click Import to open the Import page.
5. Set Certificate Type to PGP.
6. In the Alias field, enter customer-key.

✏️ Note: You must enter customer-key in this field. Otherwise, the encryption APIs can’t use this key for encrypting outbound files.

7. Click Browse to identify the location of the customer public key.
8. Click Import and Close to import the public key into the Oracle HCM Cloud keystore.

Your public key now appears on the Security Console Certificates page.

Generating the PGP Encryption Key Pair

You generate the PGP key pair on the Security Console. You download the public key to encrypt files that are inbound into HCM Cloud (for example, input data files for HCM Data Loader). To sign these inbound files, you can use your private key (customer-key_priv), which is verified using your public key (customer-key_pub) in Oracle HCM Cloud. You must have imported the customer public key.

To generate the PGP Encryption Key Pair:

1. Sign in to Oracle HCM Cloud with the IT Security Manager job role or privileges.
3. Click the Certificates tab to open the Certificates page.
4. Click Generate to open the Generate dialog box.
5. In the Generate dialog box, set Certificate Type to PGP.
6. In the Alias field, enter fusion-key.

✏️ Note: You must enter fusion-key in this field. Otherwise, the encryption APIs can’t use this key to decrypt all encrypted inbound files.

7. In the Passphrase field, enter a passphrase for the private key. This passphrase is needed when you edit, delete, or download the private key.

✏️ Note: If you forget the passphrase, then you may have to raise a service request for help to delete the private key. Once the old key is deleted, you can generate a new key using the process described here.

8. In the Key Algorithm field, select RSA.
9. In the Key Length field, select either 1024 or 2048.
10. Click **Save and Close**. The fusion-key pair is generated and ready for download. You can see the fusion-key pair on the Certificates page of the Security Console.

11. In the **Status** actions for the fusion-key pair on the Certificates page, select **Export > Public key**. Save the HCM Cloud public key (fusion-key_pub.asc) to your desktop. Use the downloaded key to encrypt files that are inbound to Oracle HCM Cloud.

### Encrypt and Upload Files Automatically

Encrypt files of data with PGP encryption and transfer them automatically between your servers and Oracle WebCenter Content using APIs and web services. For example, you can encrypt and decrypt files that contain sensitive employee data or confidential documents. You write programs to collect the encrypted files from your file server. You then place them on the Oracle WebCenter Content server and call a data loader to decrypt and load the data to Oracle HCM Cloud.

**Note:** You must set up your encryption keys before you perform these tasks.

### Supported Encryption Algorithms

Oracle HCM Cloud supports the following encryption algorithms. You must ensure that you use only supported encryption algorithms.

- Cipher: AES-128, Blowfish, CAST5, 3DES

  **Note:** These cipher algorithms aren’t supported: Twofish, IDEA, AES-192, and AES-256

- Compression: bzip2, zlib, .zip, uncompressed

- Hash: SHA-1, SHA-256, SHA-224, SHA-512, MD5, SHA-384, RIPEMD-160

### Encrypting Files

This section provides the commands to encrypt files in Microsoft Windows and Linux environments using the GnuPG encryption tool. For other tools and platforms, work with your suppliers to find the necessary commands for setting the cipher algorithm.

- Gpg4win, the official GnuPG distribution for Microsoft Windows, provides both a command-line interface and a graphical user interface for encryption, decryption, signing, and verification. For encryption, use the command-line interface. You can find Gpg4win here: [https://www.gpg4win.org/about.html](https://www.gpg4win.org/about.html).

- You can download GnuPG for Linux from various sources, depending on the Linux distribution that you’re using. Commonly used GnuPG versions can be found here: [https://www.gnupg.org/index.html](https://www.gnupg.org/index.html).

After installing the Gpg4win or GnuPG tool, follow these steps to encrypt or encrypt and sign a file:

1. Import the HCM Cloud public key (downloaded from the Security Console) using this command at the command prompt:

   ```
gpg --import <fusion public key file>
   ```

2. Perform one of these steps.
   - To encrypt a file without signing, use this command:
gpg --cipher-algo <one_of_the_supported_algorithms> -r fusion-key --encrypt <inbound_file_name>

- To both encrypt and sign a file, use this command:

  gpg --cipher-algo <one_of_the_supported_algorithms> -r fusion-key -se <inbound_file_name>

**Note:** When signing files, ensure that your private key is imported into the keystore that’s used for signing.

### Loading Encrypted Files

Perform the following steps to load encrypted files to Oracle HCM Cloud from the Oracle WebCenter Content server.

1. Write programs to send your encrypted files to Oracle WebCenter Content, using the Oracle WebCenter Content Web Services. If your home page is: https://Hostname/homePage/faces/AtkHomePageWelcome, then the Oracle WebCenterContent Server WSDL is: https://Hostname/idcws/GenericSoap?wsdl.

2. Call the loader program to pass the encryption parameter with other required parameters. The loaderIntegrationService uses the submitEncryptedBatch method, which has an additional parameter named encryptType. This parameter has the following values, which are defined in the ORA_HRC_FILE_ENCRYPT_TYPE lookup type:
   - NONE
   - PGPSIGNED
   - PGPUUNSIGNED

### Transfer Files Automatically from HCM Extracts with PGP Encryption

Transfer encrypted files to Oracle WebCenter Content using HCM Extracts and your encryption key. HCM Extracts can generate encrypted output and store it on the WebCenter Content server. For example, you can encrypt and decrypt files that contain sensitive employee data or confidential documents. Use HCM Extracts to generate encrypted files and deliver them to the WebCenter Content server. You write your own programs to collect the files.

**Note:** You must set up your encryption keys before you try to encrypt or decrypt data.

### Outbound Integrations

Set up the following information to use HCM Extracts with your outbound integrations:

1. In the Data Exchange work area, select the Manage Extract Definitions task.

2. Select the WebCenter Content delivery type on the Deliver page.

3. Enter an Integration Name. The application uses this name to create the title of the entry in WebCenter Content.

4. Select an Encryption Mode. The encryption mode is one of the values from the ORA_HRC_FILE_ENCRYPT_TYPE lookup type. It determines how the application encrypts the file before loading it to WebCenter Content. When HCM Extracts transfers the file to WebCenter Content it generates a content ID automatically with the following format: UCMFAnnnnnn.
The file includes the following properties:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author</td>
<td>FUSION_APPS_HCM_ESS_APPID</td>
</tr>
<tr>
<td>Security Group</td>
<td>FAFusionImportExport</td>
</tr>
<tr>
<td>Account</td>
<td>hcm/dataloader/export</td>
</tr>
<tr>
<td>Title</td>
<td>HEXTV1CON_{Integration Name}<em>{Encryption Type}</em>{Date Time Stamp}</td>
</tr>
<tr>
<td></td>
<td>For example: HEXTV1CON_ExtractConn1_ PGPUNSIGNED_ 17-11-2014 14-16-44</td>
</tr>
</tbody>
</table>

5. Configure the HCM Extract delivery option to output an XML (data) file directly to WebCenter Content without formatting it in BI Publisher. You can achieve this by selecting Data as the output format, omitting a template name, and selecting the WebCenter Content Delivery Type.

6. Download the encrypted files from WebCenter Content using client command-line tools or a web service call.

Decryption of Outbound Files

Using your private key, you can decrypt encrypted files that are generated from Oracle HCM Cloud. To verify signed files, you use the Oracle HCM Cloud public key. Ensure that these two keys are imported into the keystore. For both Microsoft Windows and Linux, use this command to decrypt both signed and unsigned files:

```
gpg --decrypt <inbound_file_name> --output <output_file_name>
```

Related Topics

- For more information see HCM Extracts: Changes Only (Document ID 1964920.1.) on My Oracle Support at https://support.oracle.com
39 Oracle Taleo Recruiting Cloud Service Integration

HCM and Oracle Taleo Recruiting Cloud Service Components: How They Work Together

The integration between HCM and Oracle Taleo Recruiting Cloud Service facilitates seamless flow of the recruitment process from creating a job requisition to hiring a worker.

HCM and Oracle Taleo Recruiting Cloud Service Integration

The integration between HCM and Oracle Taleo Recruiting Cloud Service involves the following stages:

1. HCM
   - Exporting work structures and worker data to Oracle Taleo Recruiting Cloud Service.

2. Oracle Taleo Recruiting Cloud Service
   - a. Creating a job requisition using the data imported from HCM
   - b. Selecting candidates for a job requisition
   - c. Hiring a candidate
   - d. Importing candidate data from Oracle Taleo Recruiting Cloud Service

3. HCM
   - a. Creating a pending worker
   - b. Converting the pending worker to an employee
   - c. Exporting changed data to Oracle Taleo Recruiting Cloud Service

Importing Data From Oracle Taleo Recruiting Cloud Service: Explained

Once a candidate is selected in Oracle Taleo Recruiting Cloud Service, you must import the candidate data to HCM to facilitate hiring the candidate as an employee or contingent worker in HCM.

Selected Candidates

When ready-to-be-hired candidates are imported from Oracle Taleo Recruiting Cloud Service you can view notifications:

- On the Selected Candidates tab of the New Workers to Process page
- In the Worklist notifications, if you are the hiring manager
Click the notification task to process the selected candidate as a pending worker. You must ensure that all information is correct before submitting the candidate for approval. Successfully processed pending workers appear on the Pending Workers tab. From that tab, you can convert them into employees.

Imported Data
The data that is imported from Oracle Taleo Recruiting Cloud Service to HCM includes the following:

- Candidate data, including:
  - Person and employment data
  - Compensation and payroll data
  - Education, certification, and previous experience details, which are captured as a part of HCM Talent
- Requisition data

Scheduled Processes
You can access the Import Oracle Taleo Recruiting Cloud Service Data process from the Scheduled Processes work area. You can run this process manually or schedule it to run at regular intervals (for example, weekly at a specified time). The likely volume and frequency of hiring persons in your enterprise determines how often you run this process.

After you run the import process, you can view the selected candidates in the New Person Overview page. The Selected Candidates page displays a list of candidates ready to be added as pending workers. Once you add a candidate as a pending worker, the candidate appears in the Pending Workers list. You can then convert the pending worker to an employee or a contingent worker. The imported person, payroll, compensation, and talent (education, certification, and previous experience) details appear by default in the Convert Pending Worker flow.

> **Note:** After you import a candidate from Oracle Taleo Recruiting Cloud Service to HCM, if you make any changes to the candidate or requisition in Oracle Taleo Recruiting Cloud Service, these changes aren't included in the subsequent import runs.

Exporting Data to Oracle Taleo Recruiting Cloud Service: Explained
Oracle Taleo Recruiting Cloud Service uses HCM work structures and worker data to create and process job requisitions. You can export this data from HCM to Oracle Taleo Recruiting Cloud Service.

HCM and Oracle Taleo Recruiting Cloud Service Mapping
The following table lists the business objects that HCM exports and the corresponding business objects that they are mapped to in Oracle Taleo Recruiting Cloud Service.

<table>
<thead>
<tr>
<th>HCM</th>
<th>Oracle Taleo Recruiting Cloud Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Departments</td>
<td>Departments</td>
</tr>
</tbody>
</table>
Employees and Users Export

As a part of the integration, HCM exports employees and users such as Managers, Hiring Managers, and Recruiters, who have the privilege to create requisitions in Oracle Taleo Recruiting Cloud Service. From the HCM perspective, user and employee data are the same, and use the same data mapping. The employee or user creation process relies on certain default user types provisioned in Oracle Taleo Recruiting Cloud Service. The following table lists the roles exported from HCM, and the user types they are mapped to in Oracle Taleo Recruiting Cloud Service.

<table>
<thead>
<tr>
<th>HCM Role</th>
<th>Oracle Taleo Recruiting Cloud Service User Type Code</th>
<th>Oracle Taleo Recruiting Cloud Service User Type Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>PER_RECRUITING_ADMINISTRATOR_JOB</td>
<td>01-RecruitingAdmin</td>
<td>Recruiting - Administrator</td>
</tr>
<tr>
<td>PER_POWER_RECRUITER_ABSTRACT</td>
<td>02-PowerRecruiter</td>
<td>Recruiting - Power Recruiter</td>
</tr>
<tr>
<td>PER_BASIC_RECRUITER_ABSTRACT</td>
<td>03-BasicRecruiter</td>
<td>Recruiting - Basic Recruiter</td>
</tr>
<tr>
<td>PER_LINE_MANAGER_ABSTRACT</td>
<td>04-PowerManager</td>
<td>Recruiting - Power Manager</td>
</tr>
<tr>
<td>PER_EMPLOYEE_ABSTRACT</td>
<td>10-Employee</td>
<td>Recruiting - Employee</td>
</tr>
</tbody>
</table>
Scheduled Processes

You can schedule the following process in the Scheduled Processes work area:

- **Export All Oracle Taleo Recruiting Cloud Service Data**
  
  Exports all work structures and worker data from HCM to Oracle Taleo Recruiting Cloud Service. You must run this process typically only once, in the beginning, to send all HCM data to Oracle Taleo Recruiting Cloud Service to enable creation of job requisitions.

- **Export Changed Oracle Taleo Recruiting Cloud Service Data**
  
  Exports any changed work structures and worker data from HCM to Oracle Taleo Recruiting Cloud Service. You can run this process manually or schedule it to run at regular intervals (for example, weekly at a specified time). The likely volume and frequency of hiring persons in your enterprise determines how often you run this process.

FAQs for Oracle Taleo Recruiting Cloud Service and HCM Integration

What are Large User Defined Selections (LUDS)?

Some business objects in HCM don’t map directly to any business objects in Oracle Taleo Recruiting Cloud Service, when exporting data from HCM to Oracle Taleo Recruiting Cloud Service. These business objects (for example, legal employers, jobs, and positions) are mapped to Large User Defined Selections (LUDS) instead. LUDS are custom, user-defined fields in Oracle Taleo Recruiting Cloud Service, specifically configured for the integration with HCM. The mappings are defined in the Oracle Taleo Recruiting Cloud Service application.
40 Web Services

Overview of Web Services

Use web services to integrate web-based applications with Oracle Human Capital Management (HCM) Cloud. Web services expose business objects and processes to other applications through the use of open standards-based technologies. For more information about web services, see:

- SOAP Web Services for Oracle HCM Cloud
- SOAP Web Services for Common Features in Oracle Applications Cloud.
Glossary

absence case
A grouping of related absences that result from the same cause used for reporting purposes.

absence category
A group of related absence types for reporting purposes.

action reason
Action reasons provide further explanation to actions, for example, an action of transfer could have reasons such as reorganization or career progression.

annualization factor
Multiplication factor used to convert base salary to an annualized amount

assignment
A set of information, including job, position, pay, compensation, managers, working hours, and work location, that defines a worker's or nonworker's role in a legal employer.

business unit
A unit of an enterprise that performs one or many business functions that can be rolled up in a management hierarchy.

Calculation Card
Captures values required for payroll calculations for some earnings and deductions, such as absence payments and involuntary deductions. For some countries, you can also create various types of cards to hold default values for tax reporting units or payroll statutory units.

compa-ratio
Salary as a percentage of salary range midpoint. 100 denotes salary at midpoint.

content item
An individual quality, skill, or qualification within a content type that you track in profiles.

content library
A repository of the content types and individual content items that can be associated with person profiles and profiles for workforce structures such as jobs and positions.
**content type**
An attribute such as a skill, quality, or qualification that is added to a profile.

**date-effective object**
An object with a change history. Professional users can retrieve the object as of a current, past, or future date.

**descriptive flexfield**
Expandable fields used for capturing additional descriptive information or attributes about an entity, such as a customer case. You may configure information collection and storage based on the context.

**development goal**
A goal that is geared toward facilitating the career growth of individuals so that they can perform better in their current job or prepare themselves for advancement.

**division**
A business-oriented subdivision within an enterprise. Each division is organized to deliver products and services or address different markets.

**document category**
A high level grouping of person documents such as visas, licenses, and medical certificates. Document subcategories provide further grouping of document categories.

**document type**
A categorization of person documents that provides a set of options to control what document information to retain, who can access the documents, whether the documents require approval, and whether the documents are subject to expiry. A document type exists for a combination of document category and subcategory.

**effective end date**
For a date-effective object, the end date of a physical record in the object’s history. A physical record is available to transactions between its effective start and end dates.

**effective sequence number**
A number that identifies the order of changes made to a date-effective object in one day. On any day, the physical record with the highest number is the current record.

**effective start date**
For a date-effective object, the start date of a physical record in the object’s history. A physical record is available to transactions between its effective start and end dates.

**element**
Component in the calculation of a person’s pay. An element may represent a compensation or benefit type, such as salary, wages, stock purchase plans, pension contributions, and medical insurance.
element entry
The record controlling an employee's receipt of an element, including the period of time for which the employee receives the element and its value.

eligibility profile
A user-defined set of criteria used to determine whether a person qualifies for a benefits offering, variable rate or coverage, compensation plan, checklist task, or other object for which eligibility must be established.

extensible flexfield
Expandable fields that you can use to capture multiple sets of information in a context or in multiple contexts. Some extensible flexfields let you group contexts into categories.

fast formula
A simple way to write formulas using English words and basic mathematical functions. Formulas are generic expressions of calculations or comparisons that repeat with different input values.

flexfield
A flexible data field that you can configure such that it contains one or more segments or stores additional information. Each segment has a value and a meaning.

flexfield segment
An extensible data field that represents an attribute and captures a value corresponding to a predefined, single extension column in the database. A segment appears globally or based on a context of other captured information.

flow
An occurrence of a flow pattern that you manage from a payroll work area or from the Data Exchange work area using the View Extracts task. The data security for your role determines which flows you can submit and access.

flow checklist
A sequence of automatic and manual flow tasks grouped into activities, such as extract reports and processes, or tasks related to payroll processing. Submitting a flow generates a checklist that you use to monitor the flow and manage its tasks.

flow pattern
A series of tasks performed in a predefined order, which are grouped into activities, such as extract reports and processes, or tasks that cover a phase of the payroll process. The flow pattern is used to generate a flow, which you can manage from its checklist.
**FTE**
Abbreviation for full-time equivalent, such as .5 for half-time work.

**global name**
A person’s name in a format and language that can be understood throughout a multinational enterprise.

**global transfer**
The transfer of a person to another legal employer.

**goal plan**
A collection of performance goals that are grouped by common characteristics, such as a specified time frame or a particular department that must work on them.

**goal plan set**
One or more performance goal plans that you assign as a group to a population set.

**grade**
A component of the employment model that defines the level of compensation for a worker.

**grade ladder**
A hierarchy used to group grades and define their sequence.

**grade rate**
Used to define pay values for grades in a legislative data group.

**grade step**
A level of increment within a grade.

**grade step progression**
The advancement of workers through a sequence of grades or steps within a progression grade ladder, according to progression rules.

**Groovy**
An object-oriented programming language for the Java Platform used as an alternative to the Java programming language. Groovy can also be used dynamically as a scripting language.

**HCM Data Loader data set**
A set of data supplied together in either compressed data (.dat) files or a spreadsheet to be uploaded to Oracle HCM Cloud.
**HR**
Abbreviation for human resource.

**job**
A generic role that is independent of any single department or location. For example, the jobs Manager and Consultant can occur in many departments.

**job family**
A group of jobs having different but related functions, qualifications, and titles. For example, you may group a trust analyst and an operations analyst into the Analyst job family.

**job role**
A role, such as an accounts payable manager or application implementation consultant, that usually identifies and aggregates the duties or responsibilities that make up the job.

**key flexfield**
Configurable flexfield comprising multiple parts or segments, each of which has a meaning either individually or in combination with other segments. Examples of key flexfields are part numbers, asset category, and accounts in the chart of accounts.

**LDAP**

**legal employer**
A legal entity that employs people.

**legislative data group**
A means of partitioning payroll and related data. At least one legislative data group is required for each country where the enterprise operates. Each legislative data group is associated with one or more payroll statutory units.

**local name**
A person’s name in a format and language that are readily understood by users in a single country but that may not be understood throughout a multinational enterprise.

**lookup code**
An option available within a lookup type, such as the lookup code BLUE within the lookup type COLORS.

**lookup type**
The label for a static list that has lookup codes as its values.

**model profile**
A collection of the work requirements and required skills and qualifications of a workforce structure, such as a job or position.
offering
A comprehensive grouping of business functions, such as Sales or Product Management, that is delivered as a unit to support one or more business processes.

organization goal
A performance goal that specifies the overall objectives of the organization.

payroll relationship
Defines an association between a person and a payroll statutory unit based on payroll calculation and reporting requirements.

payroll statutory unit
A legal entity registered to report payroll tax and social insurance. A legal employer can also be a payroll statutory unit, but a payroll statutory unit can represent multiple legal employers.

performance goal
A results-oriented goal, often using specific targets, to assess the level of a worker's achievement.

person number
A person ID that is unique in the enterprise, allocated automatically or manually, and valid throughout the enterprise for all of a person's work and person-to-person relationships.

person type
A subcategory of a system person type, which the enterprise can define. Person type is specified for a person at the assignment level.

position
A specific occurrence of one job that is fixed within one department. It is also often restricted to one location. For example, the position Finance Manager is an instance of the job Manager in the Finance Department.

progression grade ladder
A hierarchy used to group grades and define their sequence. It includes the associated rates and progression rules for each grade and step within the ladder.

progression rule
A set of criteria used to determine whether a worker is eligible to advance to a specific grade or step within a progression grade ladder.
quartile
Salary range divided into four equal parts. Quartile 1 is the lowest.

reference data set
Contains reference data that can be shared across a number of business units or other determinant types. A set supports common administration of that reference data.

role mapping
A relationship between one or more roles and one or more assignment conditions. Users with at least one assignment that matches the conditions qualify for the associated roles.

role provisioning
The automatic or manual allocation of a role to a user.

salary basis
Defines validation and payroll details for worker base pay. It identifies the currency and period of the quoted base pay and the factor used to annualize base pay. It optionally identifies components or rates used to itemize salary adjustments and the grade rate used to validate salary.

salary component
Change reasons that enable itemization of salary adjustments by entering amounts or percentages for one or more components, such as merit or cost of living adjustment.

set
Classified and grouped reference data that organizational entities share.

specialization
A logical grouping of courses aimed to help learners achieve learning goals that a single course can't achieve. A specialization contains one or more sections, which are containers for the courses.

stage table
An intermediate table to which records are loaded from a spreadsheet or file and where some validation occurs.

succession plan
A plan that identifies candidates for a role or position or to succeed a named incumbent.

succession plan owner
A person who can manage succession plans. Valid actions depend on the owner's administrator type, which can be Administrator, Candidate Manager, or Viewer.
talent pool
A selected group of workers for whom you track training, readiness, or development.

talent pool owner
A human resource specialist who can manage a talent pool.

talent review
A series of meetings where organization managers evaluate trends, assess strengths, and address areas of risk for the organization.

talent review facilitator
Human resource specialist who manages talent review meetings. A talent review meeting can have multiple facilitators.

talent review participant
Person who’s invited to attend a talent review meeting.

temporary assignment
The transfer of a person for a limited term. You terminate the temporary assignment at the end of a term, facilitating the person’s return to the source assignment.

time attribute
A qualifier associated with a time event or time entry that reflects how the time is paid, costed, billed, or recorded as an information entry. For example, the payroll time type attribute indicates whether time for payroll consumers should be paid as Regular, Overtime, or Vacation.

tree node
One of the branching points in a tree structure. It corresponds to a primary key in the view object of data.

tree version
An instance of a tree that includes life cycle elements such as start and end dates, and indicates whether the tree is active. If a tree is associated with a reference data set, all tree versions belong to one set.

URL
Abbreviation for Uniform Resource Locator.

view object
A component that simplifies querying and working with business object rows.

work relationship
An association between a person and a legal employer, where the worker type determines whether the relationship is a nonworker, contingent worker, or employee work relationship.
**worker time entry profile**
A collection of layout rules and specifications that determine the time card appearance and control when employees can take action on their time cards.

**worker time processing profile**
A collection of the time card period and the time entry and time calculation rule sets for both the employee and the time consumer.

**XML**
Abbreviation for Extensible Markup Language.