

# **Oracle Enterprise Taxation Management**

User Documentation

Version 2.2.0

**E17777-01**

March 2009

Copyright © 2000, 2009, Oracle and/or its affiliates. All rights reserved.

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

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

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

#### U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007).

Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

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

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Taxpayer Information

We use the term **taxpayer information** to reference the demographic, geographic, and financial objects that form the core of your system. In this section, we describe how to maintain these objects.

## Contents

- [Understanding the Account Model](#)
- [Navigating The Account Model Using Control Central Dashboard Portal](#)
- [Maintaining The Account Model](#)
- [Addition Location Management Tools](#)
- [Setting Up Bill Print Groups](#)
- [The Big Picture Of Customer Contacts](#)
- [Printing Letters](#)

## Understanding the Account Model

---

The "Account Model" consists of four objects that form the core of the system: **Person**, **Account**, **Location**, and **Obligation**. These objects hold demographic, geographic, and financial information about your taxpayers.

You must intuitively understand the concepts embodied in the "Account Model" before you can perform the business processes in the system. In this section, we provide an overview of these objects. In later sections, we describe how you maintain this information.

## Contents

- [Persons](#)
- [Accounts](#)
- [Tax Roles](#)
- [Obligations](#)
- [Locations](#)
- [Putting It All Together](#)
- [Sample Account Model For An Individual Taxpayer](#)
- [How To Set Up Taxpayer Hierarchies](#)

## Persons

A person exists for every individual or business with which your company has contact. Besides taxpayers, persons exist for contractors, accountants at corporate taxpayers, third party guarantors, collection agencies, etc.

On a person is maintained demographic information like Name, Mailing Address, Phone Numbers, Email Address, etc.

Most persons are linked to at least one [account](#) because, without an account, the person cannot have financial interactions with the tax authority.

Refer to <a href="#">Maintaining Persons</a> for more information about persons.
--

## Accounts

Accounts are the entities for which financial interactions happen. These can include, returns, bills, and payments. You must create at least one account for every taxpayer. The account contains information that controls how returns, bills, and payments are created and handled.

Every account must reference at least one [person](#) because the person contains the taxpayer's demographic information (e.g., names, phone numbers, forms of ID). We refer to this individual as the "main" person linked to the account. In addition to the "main" person, an account may reference other types of persons, e.g., the billing contact, power of attorney, accountant, etc.

Most accounts are linked to at least one [tax role](#) because the tax role captures information related to a specific tax type that is applicable to the account.

Most accounts are linked to at least one [obligation](#) because, without an obligation, there is no ability to assess a taxpayer. An account without an obligation may exist; you just won't be able to do much in the system with such an account.

Refer to [Maintaining Accounts](#) for more information about accounts.

## Tax Roles

Tax Roles represent the ongoing obligations of a given tax type within an account at a given point in time. It may also be thought of as the reason a taxpayer must interact with the tax authority, or one of the taxpayer's relationships with the tax authority.

Tax roles include the dates that the tax is effective. If you consider a business, some tax types are required for the entire time a company is in business, such as corporate income tax. Other tax types may only be in effect if the company is engaging in certain activities, which may change over time.

The tax role often governs the filing calendar that dictates the filing periods for which the taxpayer is obliged to file. For every filing period that the taxpayer is expected to file a return for this tax type, an appropriate [obligation](#) must be defined for this account and tax role.

Refer to [Maintaining Tax Roles](#) for more information about accounts.

## Obligations

Think of an obligation as a contract between the tax authority and the taxpayer. For example, a corporate entity has certain contractual responsibilities between themselves and the tax authority. These responsibilities are tracked through the obligation. Some tax authorities refer to an obligation as a filing period or tax period. Every [account](#) should have at least one obligation (otherwise, the account has no financial obligations with your tax authority). There is no limit to the number of obligations that may be linked to an account.

Every tax role should have at least one obligation. Note that obligations are not required to have a tax role. Configuration on the obligation type's tax type controls whether a tax role is required.

Refer to [Maintaining Obligations](#) for more information.



## Locations

A location is created for every real property location. On a location is maintained geographic information like:

- The address.
- Characteristics that determine tax jurisdictions.
- Descriptions of unusual situations associated with a property.
- The list goes on...

Refer to [Maintaining Locations](#) for more information about locations.

## Putting It All Together

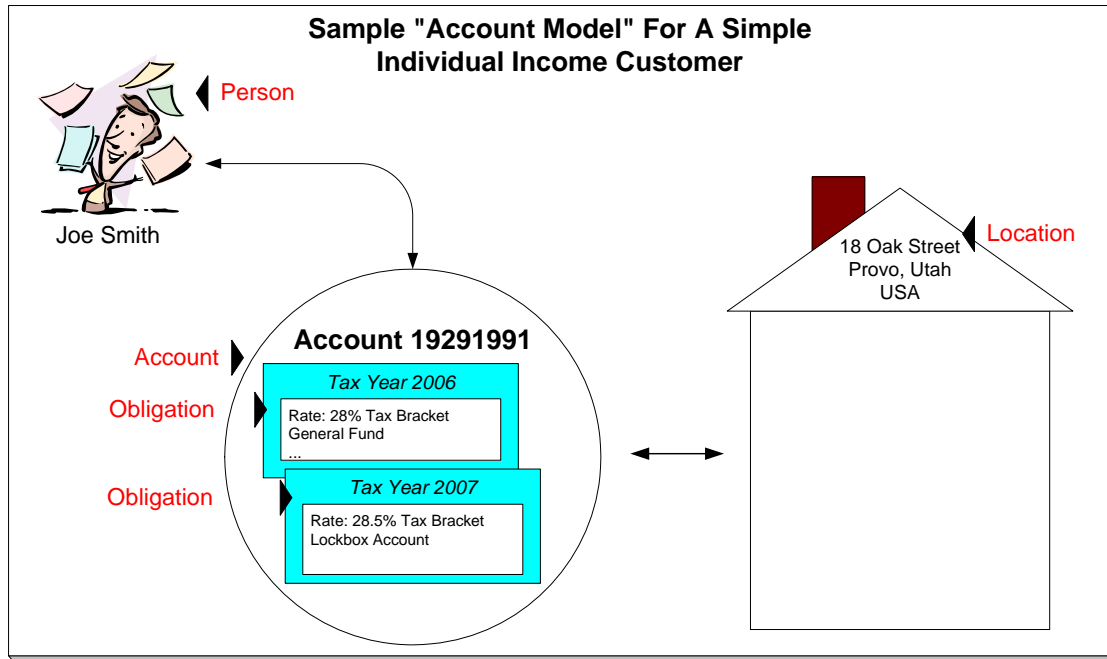
When a new taxpayer registers, you create both person and account objects for the new taxpayer.

A business with a single location uses the same “Account Model” objects as an individual income taxpayer. If the business expands and opens at new locations, you simply link new obligations to the existing account. You don't have to redefine person or account information.

A person and an account are needed for every taxpayer.

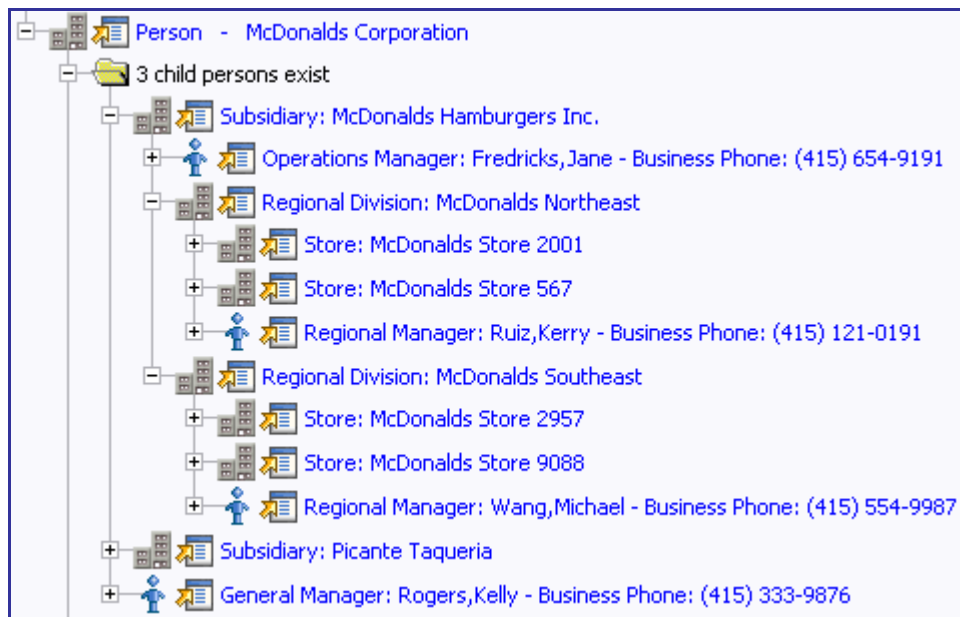
## Sample Account Model For An Individual Taxpayer

The following illustration shows a simple residential taxpayer's “account model” objects.



## How To Set Up Taxpayer Hierarchies

Consider the following taxpayer hierarchy:



The above hierarchy shows a parent company (**McDonalds Corporation**) with two subsidiaries and one general manager. Each subsidiary, in turn, has subsidiaries and these subsidiaries can have their own subsidiaries ...

You must set up a person for every individual and business in a hierarchy. After the basic demographic information is defined, you define each person's children using the [Person to Person](#) page.

**A script can be used to set up a hierarchy.** The demonstration database contains a [script](#) that guides users through the process of setting up a hierarchy. Please speak to your implementation staff if you'd like to [import](#) this script for use in your implementation.

After a hierarchy is set up, the system displays its members in the [Person Tree Zone](#) (this is a zone on the [Taxpayer Information Portal](#)). In addition, if the persons in a hierarchy have accounts, summary information about these accounts is displayed in the [Account Summary Zone](#).

While hierarchies would typically be set up to define parent companies and subsidiaries, you can use this functionality for many other purposes. For example, a taxpayer relationship manager could set up a hierarchy containing their "important" taxpayers. To do this, they'd need to create a "dummy" person for the parent and then link the "important" taxpayers to it. After this hierarchy is set up, the manager could see a summary of these taxpayers in the [Account Summary Zone](#) whenever they display the "dummy" person.

## Navigating The Account Model Using Control Central

---

Control Central is the name given to the functionality that helps you:

- Find a person.
- Find an account.
- Find a location.
- View information about a person / account / location.
- View alerts and messages about a person / account / location.
- Navigate to pages that contain information related to a person / account / location.

### Contents

- [Control Central - Main](#)
- [Control Central - Account Information](#)
- [Control Central - Taxpayer Information](#)
- [Control Central - Account Tree](#)
- [Control Central - Location Tree](#)
- [Control Central - Bill/Payment Tree](#)
- [How To Add A New Taxpayer From Control Central](#)

## Control Central - Main

Open **Control Central, Main** to find a taxpayer and/or a location.

**Automatic transfer to Account Information tab.** You are automatically transferred to [Control Central - Account Information](#) when an account / location is selected on this page.

The contents of this section describe how to use this page.

## Contents

[Search Facilities](#)  
[Search Options](#)  
[Wild Cards](#)  
[Search Results](#)

## Search Facilities

The top of Control Central is where you enter the criteria used to find a person, account or location.

**Multiple search criteria.** You can search for a taxpayer using a combination of **Name**, **Address**, **City**, and **Postal Code**. When multiple fields are populated, the system searches for taxpayers that match all such criteria. For example, if you enter a **Name** of **Smith**, and a **Postal** code of **94114**; only taxpayers named **Smith** with service in postal area **94114** will be displayed.

The following table describes each of the different search methods.

Search Method	Description
Name	<p>Use this field to search for a person or account using a person's name. The person may be the main taxpayer on the account or a third party related to the account.</p> <p>You can enter all or part of the person's name. The name search is not case sensitive.</p> <p>If you need to add a new taxpayer, use either of the following options:</p> <ul style="list-style-type: none"> <li>- Click the + button. Clicking this button opens the <a href="#">Person - Main</a> page where you can add a new person. Refer to <a href="#">How To Add A New Taxpayer From Control Central</a> for more information about how to add a new taxpayer (i.e., Person and Account) from this page.</li> </ul>
Address	<p>Use this field to search for a location or account using the first line of a service address.</p> <p>You can enter all or part of the first line of the address. For example, if you're looking for 324 Hawthorne Lane, you can enter <b>324 Haw</b> rather than the entire address.</p> <p>The address search is not case sensitive.</p>
City	<p>Use this field to search for a location or account using a city name associated with a service address.</p> <p>You can enter all or part of the city name. For example, if you're looking for someone in San Francisco; you can enter <b>San Fra</b>, <b>San F</b>, <b>San Francisco</b>, etc.</p> <p>The city search is not case sensitive.</p> <p>We strongly recommend qualifying a <b>City</b> search with some other information (e.g., <b>Name</b>) otherwise the resultant list can be too large to deal with.</p>
Postal	<p>Use this field to search for a location or account using a postal code associated with a service address.</p> <p>You can enter all or part of the postal code. For example, if you're looking for someone in 94114; you can enter "94114", "9411", etc.</p> <p>We strongly recommend qualifying a <b>Postal</b> search with some other information (e.g., <b>Name</b>) otherwise the resultant list will be too large to deal with.</p>

Search Method	Description
Account ID	<p>Use this field to search for an account using an account number. After entering this information, click the search button adjacent to the Account ID entry field (or press Enter) to search for an account. The <b>Account ID</b> cannot be combined with other search fields.</p> <p>You can enter all or part of the account ID. For example, if you're looking for account 1929119291; you can enter "1929119291", "192911", etc.</p> <p>If you need to add a new account for an existing person, click the + button. Clicking this button opens the <a href="#">Account - Main</a> page on which a new account can be added.</p>
Phone Format/Number	<p>Use this field to search for an account using one of the telephone numbers linked to one of the persons linked to the account. The person may be the main taxpayer on the account or a third party related to the account.</p> <p>You may enter the phone number in the format indicated or you can enter the information as a contiguous string of numbers and the system will format this for you.</p> <p>Wild card searches are not supported. However, you can enter the first few digits of the phone number and the system will look for all accounts with these digits.</p> <p>After entering this information, click the search button (or press Enter) to search for an account linked to a person with this telephone number. The system displays every account with the input telephone number in the search results.</p>
Person ID Type/Value	<p>Use these fields to search for a person or account using one of the ID's linked to one of the persons linked to the account. You must enter both an identifier type and number.</p> <p>You may enter the person's ID in the format dictated by the identifier type (if any). Please note that if the <b>ID Number</b> should be formatted (e.g., dashes in an American social security number), you do not have to enter the dashes. Rather, you can enter the information as a contiguous value and the system will format this for you.</p> <p>Wild card searches are not supported. However, you can enter the first few digits of the ID number and the system will look for all persons and accounts with these digits.</p> <p>After entering this information, click the search button (or press Enter) to search for a person / account linked to a person with this ID. The system displays every person with the input ID in the search results.</p>
Geo ID Type/Value	<p>Use these fields to search for a location using one of the geographic values linked to the <a href="#">location</a>. You must enter both a geographic type and value.</p> <p>Wild card searches are not supported. However, you can enter the first few digits of the geographic value and the system will look for all locations with these digits.</p> <p>After entering this information, click the search button (or press Enter) to search for a location with this ID. The system displays every location with the input ID in the search results.</p>

**Automatic transfer to Account Information tab.** If your search criteria result in a single taxpayer being retrieved, you are automatically transferred to [Control Central - Account Information](#) with that taxpayer's information displayed.

## Search Options

The **Show All Locations** search option controls whether you see all of an account's locations in the [search results](#). If turned on, every location linked to an account is shown in the search results. If turned off, only one location is shown per account (and this location is selected at random).

This switch pertains to the following search methods:

- Search by name (and all address constituents are left blank)
- Search account ID
- Search by person phone number
- Search by person identifier

This switch does NOT pertain to the following search methods as these are location-oriented, not account-oriented:

- Search by name when any address constituent is entered
- Search by address constituent (when name is blank)
- Search by geographic identifier

The benefit of turning this switch off is that an account will be immediately selected when you enter unique account-oriented search criteria. For example, if you specify an **Account ID** and press enter, the system automatically selects the account; you don't have to wait for the search results to be populated with the multitude of locations linked to the account (and then select one of the rows). Please note, an alert highlights if the selected account has other locations.

**Default note.** The first time you open this page, this switch defaults from your [user preferences](#). When you return to this page, the last value of this switch is defaulted.

## Wild Cards

The control central searches against **Name**, **Address**, and **City** support wild cards. The wild card character is **%** and represents any number or character. The system always appends a **%** for you when you use these search methods. Examples will help clarify this functionality:

- If you enter a **B**, the system searches for **B%** (and will find all objects that begin with **B**).
- If you enter **B%N**, the system searches for **B%N%**. It will find anything that
  - Starts with a **B**, and
  - has zero or more other characters followed by an **N**.
- If you enter **%B** the system searches for **%B%**. It will find anything with a **B** in it, no matter what it starts and ends with.

**Certain wildcard searches can result in lengthy response times.** If you use the wildcard character to prefix your search criteria (e.g., **%San Fran**), lengthy response times can result. Why? Because the system will have to look at EVERY taxpayer in the system before it can return the results.

## Search Results

The search results at the bottom of the page contain persons, accounts, and locations that match your search criteria. The information displayed in this area differs depending on the type of search you perform.

**The first X matches are returned.** The number of taxpayers that are returned is limited by the system's memory buffers. If you do not find the account you are looking for in the set of results, refine your search criteria.

**Automatic selection if only one match.** You don't have to select the desired person / account / location if there is one and only one object that matches your search criteria; the system automatically selects it and transfers you to [Control Central - Account Information](#).

**Use Tab and Enter to select an account.** You don't have to use the mouse to select an account when multiple matches are returned. Instead, the system highlights the first row in the search results. If this is the taxpayer you want, press **enter** to select it. If not, press **tab** until you've highlighted the desired taxpayer (and then press **enter** to select it).

**Use next and previous buttons to step through a list of taxpayers.** When you select an account from the search results, you are automatically transferred to [Control Central - Account Information](#). If you want to look at a different account that appeared in the search results, you do NOT have to return to the **Main** tab. Rather, you can click the [next in list](#) and [previous in list](#) buttons to scroll up and down through the search results (note that there are accelerator keys for each of these buttons so you don't have to use the mouse to take advantage of this feature).

## Control Central - Account Information

Once [Control Central - Main](#) has populated an account, you are brought to this page to see an overview of the account.

**Global context.** Selecting an account on control central causes the [global context](#) information to be refreshed. Various zones available on this page use the value in the global context to display relevant data for the appropriate account, person and location.

**User configurable.** Refer to [Each User Can Customize Which Zones Appear](#) for information about how to configure which zones appear. The image of the page below contains a limited number of the possible zones.

### Description of Page

**Navigation hint.** The [Go To Control Central](#) option on the account context menu navigates to this tab page.

**A mouse with a roller is useful.** This page can extend vertically past the normal desktop boundary. You will find that a mouse with a “roller” will facilitate navigating through the page.

The contents of this section describe the zones that are available on this portal page.

### Contents

- [Account Activity History Zone](#)
- [Account Financial History Zone](#)
- [Alert Zone](#)
- [Bill Graph Zone](#)
- [Context Zone](#)
- [Credit Allocation Detail Zone](#)
- [Collections Information Zone](#)
- [Taxpayer Information Zone](#)
- [Financial Information Zone](#)
- [Location Information Zone](#)
- [Timeline Zone - Account Info](#)

### Account Activity History Zone

The **Account Activity History Zone** is a grid that summarizes a variety of account-related events (in reverse chronological order). Pushing the button adjacent to the information transfers you to an appropriate page. The following table lists the type of information that may appear in this zone and the page to which you will be transferred if you push the adjacent button.

Activity Label	What Is Displayed	Drill Down Transfers You To
<b>Overdue Event.</b> <i>One row is displayed for every <b>Pending</b> overdue event linked to the account.</i>	The overdue event's standard information string (this is dynamic as it's constructed by an algorithm)	Overdue Process - Events
<b>Overdue Process.</b> <i>One row is displayed for every <b>Inactive</b> overdue process linked to the account. <b>Inactive</b> processes are displayed (as opposed to <b>active</b>) because <b>active</b> overdue processes have <b>pending</b> events and <b>pending</b> events are shown separately.</i>	The overdue process's standard information string (this is dynamic as it's constructed by an algorithm)	Overdue Process - Main
<b>Customer Contact.</b> <i>One row is displayed for every customer contact associated with the main taxpayer linked to the account.</i>	The main name of the person associated with the customer contact, the customer contact's class and type, and the customer contact's date	Customer Contact - Main
<b>Compliance Rating History.</b> <i>One row is displayed for every <a href="#">compliance rating transaction</a> that affects the account's compliance rating</i>	The effect on the account's compliance rating, the end date, creation date and the process that created the transaction.	Account - Collections
<b>Case.</b> <i>One row is displayed for every case associated with the account.</i>	The case's standard information.	Case - Main

**Warning!** The first 25 rows are displayed when this zone is initially built. If more rows exist, a **Get All** button appears. If you click this button, the system retrieves a maximum of 750 rows.



## Account Financial History Zone

The **Account Financial History Zone** lists an account's financial events in reverse chronological order. You can use this grid to both view high-level information about these events and to transfer to the respective page in which an object is maintained.

**Approximately a year and half is shown.** This zone shows all financial transactions within 20 months of the latest financial transaction. This limitation exists to prevent this zone from becoming unmanageably long.

The following columns are displayed in the grid:

<b>Effective Date</b>	This is the date the event starts aging. This column will be blank if the FT has not started aging yet.
<b>Financial Transaction Type</b>	This column indicates the type of financial event: <i>Bill</i> , <i>Payment</i> , <i>Bill Cancellation</i> , <i>Pay Cancellation</i> , <i>Adjustment</i> and <i>Adjustment (Cancel)</i> . If the event is related to an adjustment, the adjustment type's description is displayed instead of "Adjustment".
<b>Current Amount</b>	This column shows the financial event's effect on the account's current balance.
<b>Current Balance</b>	This column shows the account's current balance after the financial event.
<b>Payoff Amount</b>	This column shows the financial event's effect on the account's payoff balance. <b>Payoff Amount</b> will be dimmed if it is the same as its <b>Current Amount</b> .
<b>Payoff Balance</b>	This column shows the account's payoff balance after the financial event. <b>Payoff Balance</b> will be dimmed if it is the same as its <b>Current Balance</b> .

If you need to see more information about a specific financial transaction, press the go to button to transfer to the respective page in which the information is maintained.

For information about current and payoff balance, refer to [Current Amount versus Payoff Amount](#).

**Warning!** The first 25 rows are displayed when this zone is initially built. If more rows exist, a **Get All** button appears. If you click this button, the system retrieves a maximum of 750 rows.

## Alert Zone

The **Alert Zone** is a grid that contains messages highlighting a variety of situations. Clicking on the hyperlink transfers you to an appropriate page. The following table lists the various alerts that may appear and the page to which you will be transferred if click on the hyperlink. The table lists the alerts in alphabetical order, but it includes a column indicating the order in which the alert will appear.

**You can create additional alert conditions.** The following table contains those alerts supported in the base package. It is possible to highlight additional conditions by plugging-in the appropriate algorithm(s) on the [installation record](#). Refer to [Count Customer Contacts of a Given Type](#) and [Count Pay Plans with a Given Status](#) for examples of such algorithms.

**Account alerts are implementation-specific.** On [Account - Alerts](#), a user can define account-specific alerts that should be displayed whenever the account is selected. The [Alert Types](#) control table contains the possible alert messages.

Alert Text	Order	Description	Drill Down Transfers You To
<i>Account alerts</i>	4	Appears for each <a href="#">user-defined alert</a> associated with an account.	Account - Alert
Account has Multiple Locations	12	Appears when the account has Obligations that are linked to different locations	Control Central - Account Tree
Auto-Pay Active: <i>auto pay method description</i>	13	Appears when the account has an <a href="#">automatic payment</a> option effective on the current date. The auto pay method description (i.e. <i>Direct Debit</i> or <i>Payment Advice</i> ) appears only if payment advice functionality is enabled. Refer to <a href="#">Payment Advices</a> for more information on payment advice functionality.	Account - Auto Pay
Collection Referral Active	5	Appears when the account has active collection agency referrals.	Collection Agency Referral
Comment Exists On Account	3	Appears when a freeform comment has been entered for the account (on <a href="#">Account - Main</a> )	Account
Compliance rating: NNNN	6b	Appears when the account's <a href="#">compliance rating</a> falls below the <a href="#">compliance rating threshold</a> on the installation table. Refer to <a href="#">How are compliance rating transactions created</a> for information describing how a taxpayer's compliance rating is affected by various system and user events.	Account - Collections
<i>Installation dynamic alerts</i>	16	You can install plug-in algorithms on the installation record that will create alerts (if given situations exist). Refer to <a href="#">Count Customer Contacts of a Given Type</a> and <a href="#">Highlight Active Overdue Processes</a> for examples of such algorithms	<i>Varies by plug-in</i>
Last Contact: <i>days old - user name who added the contact</i>	1	Appears when the person has a <a href="#">customer contact</a> . The "days old" presents how old the contact is (based on the contact's date). The word <i>Today</i> is shown if the last contact was on the current date. The word	Customer Contact

		<i>Yesterday</i> is shown if the last contact was on the current date - 1 day. If neither is applicable, the number of days old is shown.	
Location has n Child Location (s)	17	Appears when the <a href="#">location</a> is defined as the parent location for one or more locations. Refer to <a href="#">Define Location Hierarchy</a> for more information.	Location Management
Location is linked to Parent Location	16	Appears when the <a href="#">location</a> defines a parent location. Refer to <a href="#">Define Location Hierarchy</a> for more information.	Location
Multiple Financially Resp	14a	Appears when the account has more than one financially responsible <a href="#">person</a> linked to it	Control Central - Account Tree
<i>Obligation type alert message</i>	7	Appears when the taxpayer has an obligation with an obligation type that has been marked with an alert message (refer to Obligation Type - Details)  Note that if the account in control central has more than one obligation, pressing the drill down button will also launch the obligation search dialog.	Obligation
<i>Obligation type description and status of a pay plan</i>	8	Appears for every <i>pending start</i> , <i>pending stop</i> , and <i>active pay plan</i> that is linked to the account.	Pay Plan
n Open Contact(s) for Person	2	Appears if there are <i>open</i> customer contacts associated with the person. Refer to <a href="#">Customer Contacts Can Be Used As Case Files</a> for more information.	Customer Contact
Person is linked to Multiple Accounts	10	Appears when the person has multiple accounts. A person's accounts are maintained on the <a href="#">Account - Person</a> page.	Control Central
n Persons on Account	14c	Appears when the account has multiple persons linked to it. An account's persons are maintained on the <a href="#">Account - Person</a> page.	Account - Person
Reactivated Obligations Exist	9	Appears when any of the account's obligations exist in the state of <i>reactivated</i> (i.e., a financial transaction has been generated after the obligation was <i>closed</i> ).	Obligation
Seasonal Address Exists	15b	Appears if the account has a seasonal address and there is no seasonal address active on the current date.	Person - Correspondence Info
Seasonal Address Currently Effective	15a	Appears if the account has an active seasonal address that is effective on the	Person - Correspondence Info

(MM/DD - MM/DD)		current date. The date range that the seasonal address is effective is included in the alert.	
Third Party Guarantor	14b	Appears when the account has a third party guarantor. An account's third party guarantors are maintained on the <a href="#">Account - Person</a> page.	Control Central - Account Tree

## Bill Graph Zone

The **Bill Graph Zone** illustrates the ending balance and new charges on each of the taxpayer's historical bills.

**Approximately a year and half is shown.** This zone shows all bills within 20 months of the latest bill. This limitation exists to prevent this zone from becoming unmanageably wide.

For balance forward accounts, both ending balances and current charges are shown in the graph. For [open-item accounts](#), only the current charges are shown in the graph.

The following points describe unique features of this zone:

- If you leave the mouse pointer stationary on a bar, summary information about the bill will be displayed.
- If you click on a bar, you will be transferred to the bill page with the respective bill displayed.

## Context Zone

The **Context Zone** contains basic information about the Person / Account / Location on which you are working.

### Person

This part of the context area contains the person's standard information. Note, this information is formatted by a plug-in algorithm on the [installation record](#). Refer to the base package's [person information algorithm](#) for an example. If you prefer different formatting logic, your system administrator should configure the system appropriately.

### Account ID

This part of the context area contains the unique identifier of the account and the name of its main taxpayer.

Adjacent to the account ID appears a "check digit". This is for information purposes only and is not needed to operate the system. Refer to the [Description of Page](#) section under [Account - Main](#) for a description of how "check digit" is calculated.

Below **Account ID**, the account's **Current Balance** and **Payoff Balance** are displayed. **Payoff Balance** will be hidden when it's the same as the **Current Balance**. Refer to [Current Amount versus Payoff Amount](#) for more information.

### Location

If the account has an obligation linked to a characteristic location, the location address is displayed. Note, the address information is formatted by a plug-in algorithm on the [installation record](#). Refer to the base package's [location format algorithm](#) for an example. If you prefer a different format, your system administrator should configure the system appropriately

## Credit Allocation Detail Zone

The Credit Allocation Detail zone shows the results of [dynamic credit allocation](#) of an account or one of its obligations.

The default is to show the credit allocation detail for all obligations on the account that support dynamic credit allocation. Use the **Obligation** dropdown and click **Calculate** to restrict the information for a specific obligation.

An obligation that supports dynamic credit allocation must reference a [Determine Detailed Balance algorithm](#).

A message above the grid indicates whether the data displayed in the grid is for the account or an obligation. If the data displayed is for an obligation, the "Calculated through" information indicates the last time that penalty and interest was calculated. If the display is for the account and the obligations have different Calculate Through dates, the message indicates this.

The grid displays the following rows:

- A row for each [debt category](#) for which the account / obligation has financial transaction posted.
- A row for **Unapplied Credits** displays if the overall balance is a credit
- A row for the Total displays with the total of each column

The grid displays the following columns:

- **Amount Assessed** indicates the original amount of each debt category. This is a sum of the amount of each financial transaction that qualifies for the row.
- **Amount Collected** indicates the total amount payments that have been allocated to one of the debt category rows.
- **Amount Waived** indicates the total amount of any [waivers](#) that have been applied to one of the debt category rows. This should only appear for a "penalty" or "interest" type of debt category.
- **Amount Written Off** indicates the total amount of any amount written off. Written off amounts are identified by looking at credit adjustments where the adjustment category indicates that it is a **write off**.

- **Other Credit** indicates the amount of any other type of credit that has been "applied" to this debt category.
- **Total** is a total for the amounts in the row.

All the amount fields are hypertext. Clicking on the amount fields will cause a list of all financial transactions that contribute to that amount to appear.

### Collections Information Zone

The **Collections Information Zone** is a grid that contains a variety of collections-oriented events. Pushing the button adjacent to the information transfers you to an appropriate page. The following table lists the type of information that may appear in this zone and the page to which you will be transferred if you push the adjacent button.

Activity Label	What Is Displayed	Drill Down Transfers You To
<b>Overdue Process.</b> <i>One row is displayed for every <b>Active</b> overdue process linked to the account.</i>	The overdue process's standard information string (this is dynamic as it's constructed by an algorithm)	Overdue Process - Main
<b>Pay Plan.</b> <i>One row is displayed for every <b>Active</b> pay plan linked to the account.</i>	The pay plan standard information string (this is dynamic as it's constructed by an algorithm)	Pay Plan - Main

**Warning!** The first 25 rows are displayed when this zone is initially built. If more rows exist, a **Get All** button appears. If you click this button, the system retrieves a maximum of 750 rows.

### Taxpayer Information Zone

The **Taxpayer Information Zone** is a grid that contains information about the current person and account. Pushing the button adjacent to the information transfers you to an appropriate page. The following table lists the type of information that may appear in this zone and the page to which you will be transferred if you push the adjacent button.

**Note.** Rows are suppressed if the related data is blank for the person / account.

Label	What Is Displayed	Drill Down Transfers You To
Account ID	The account's unique identifier (i.e., account ID) is displayed	An account <a href="#">context menu</a> has been provided (allowing you to drill to a variety of account-oriented pages)
Primary Taxpayer	The main taxpayer's name. Note, the person's name is formatted by a plug-in algorithm on the <a href="#">installation record</a> . Refer to the base package's <a href="#">name format algorithm</a> for an example. If you prefer different formatting logic, your system administrator should configure the system appropriately	A person <a href="#">context menu</a> has been provided (allowing you to drill to a variety of person-oriented pages)
Set Up Date	The account's setup date	Account - Main

<b>Division</b>	The account's division	Account - Main
<b>Account Type</b>	The account's account type	Account - Main
<b>Bill Cycle</b>	The account's bill cycle	Account - Main
<b>Current Compliance Rating</b>	The account's current compliance rating	Account - Collections
<b>Next Compliance Review Date</b>	The next date on which the account's debt will be reviewed by the account debt monitor.	Account - Collections
<i>A separate row is displayed for each characteristic linked to the account. Each row's label is the characteristic type's description.</i>	The characteristic's value	Account - Characteristic
<b>Auto Pay Source Code</b>	The account's auto pay source	Account - Auto Pay
<i>A separate row is displayed for each miscellaneous person linked to the account. Each row's label is the person relationship type's description.</i>	The main name of the miscellaneous person	A person <a href="#">context menu</a> has been provided (allowing you to drill to a variety of person-oriented pages)
<i>A separate row is displayed for each characteristic linked to the person. Each row's label is the characteristic type's description.</i>	The characteristic's value	Person - Characteristic
<i>A separate row is displayed for each telephone number linked to the person. Each row's label is the phone type's description.</i>	The telephone number	Person - Main
<i>A separate row is displayed for each identifier linked to the person. Each row's label is the ID type's description.</i>	The person identifier value	Person - Main
<b>Email Address</b>	The person's email address	Person - Miscellaneous

**Warning!** This zone does not support [data masking](#). If your implementation masks any of the information that appears on this zone, you must disable it and implement a zone that will mask your data appropriately. The demonstration system provides a sample zone that your implementation can import and then customize if necessary; this zone's code is **CL\_CIMAP**. To disable the base package zone simply deny access to its associated application service. You should only grant access to your new Taxpayer Information zone.

## Financial Information Zone

The **Financial Information Zone** is a grid that contains financial information related to the account. Clicking the hyperlink transfers you to the appropriate page. The following table lists the type of information that appears in this zone and the page to which you are transferred if you click the adjacent icon.

**Note.** Rows are suppressed if the related data is blank for the person / account.

Label	What Is Displayed	Drill Down Transfers You To
Current Balance	The account's current balance.	<i>Not applicable</i>
Payoff Balance	The account's payoff balance. It will only be displayed if the amount differs from the current balance.	<i>Not applicable</i>
Last Bill	The date and amount of the account's last bill along with its due date.	Bill - Main
Last Payment	The date and amount of the account's last payment.	Payment Event - Main
Previous Bill	The date and amount of the bill prior to the last bill for the account.	Bill - Main
Next Bill Date	The next date on which a bill is scheduled to be produced for the account (based on the account's bill cycle's schedule)	<i>Not applicable</i>
Pending Bill Exists	The date of the pending bill. This row only appears if there is a pending bill associated with the account	Bill - Main
Freezable Bill Segments	If <i>freezable</i> bill segments exist, this row contains the number of segments and their total amount.	Bill Segment - Main
Incomplete Adjustments	If <i>incomplete</i> adjustments exist, this row contains the number of adjustments and their total amount.	Adjustment - Main
Freezable Adjustments	If <i>freezable</i> adjustments exist, this row contains the number of adjustments and their total amount.	Adjustment - Main
Payment(s) Pending Upload	Appears when the account has a <i>pending</i> payment staging record (i.e., the account has an external payment that has not been uploaded into the system). To prevent payments from being created and frozen before the actual payment is received, external payments are not uploaded into the system until their accounting date is reached.  Note that the amount displayed represents the amount tendered for the account, but does not reflect how the payment may be distributed (i.e., the payment may be distributed to other accounts).  For information about payment staging records, refer to <a href="#">Interfacing Payments from External</a>	Payment Upload Staging



	<a href="#">Sources.</a>	
Incomplete Payment(s)	Appears when the account has payments that are <i>incomplete</i> . This row contains the number of payments and their total amount. Note, <i>incomplete automatic payments</i> that are pending distribution are not included in this alert as they are highlighted on the <b>Auto Pay will be distributed on</b> described below.	Payment - Main
Payment(s) with Errors	Appears when the account has payments that are in <i>error</i> . This row contains the number of payments and their total amount.	Payment - Main
Freezable Payments)	If <i>freezable</i> payments, this row contains the number of payments and their total amount.	Payment Event - Main
Auto Pay will be created on	This row shows the date that the next auto pay will be created. This alert will appear after a bill is completed for an account on auto pay. Refer to <a href="#">How And When Are Automatic Payments Created?</a> for more information.	Bill - Main
Auto Pay will be distributed on	This row shows the date that the next auto pay will be distributed. This alert will appear after a bill is completed for an account on auto pay and the automatic payment has been created/extracted. Refer to <a href="#">How And When Are Automatic Payments Created?</a> for more information.	Payment Event - Main

### Location Information Zone

The **Location Information Zone** is a grid that contains information about the current location. Pushing the button adjacent to the information transfers you to an appropriate page. The following table lists the type of information that may appear in this zone and the page to which you will be transferred if you push the adjacent button.

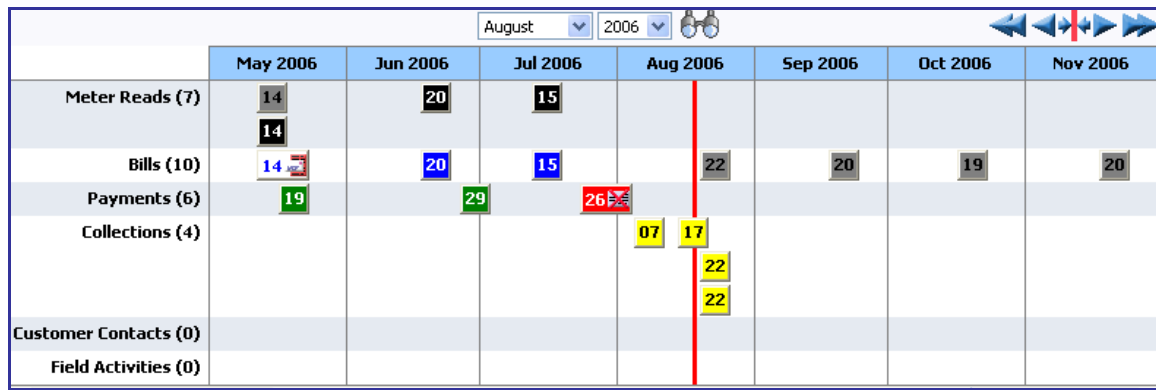
**Note.** Rows are suppressed if the related data is blank for the location.

Label	What Is Displayed	Drill Down Transfers You To
Location Information	The location's address. Note, the address information is formatted by a plug-in algorithm on the <a href="#">installation record</a> . Refer to the base package's <a href="#">location format algorithm</a> for an example. If you prefer a different format, your system administrator should configure the system appropriately	A location <a href="#">context menu</a> has been provided (allowing you to drill to a variety of location-oriented pages)
Division	The location's division	Location - Main
<i>A separate row is displayed for each</i>	The characteristic's value	Location - Characteristic

<i>characteristic linked to the location. The label in each row is the respective characteristic type's description.</i>		
--	--	--

## Timeline Zone - Account Info

Timeline zones show when significant events have occurred in the past and when significant events will occur in the future. For example, a timeline can show when payments and bills have been received for a taxpayer. The following is an example of a timeline zone:



Sample Timeline Zone

The topics in this section describe the rich functionality available in timeline zones.

## Contents

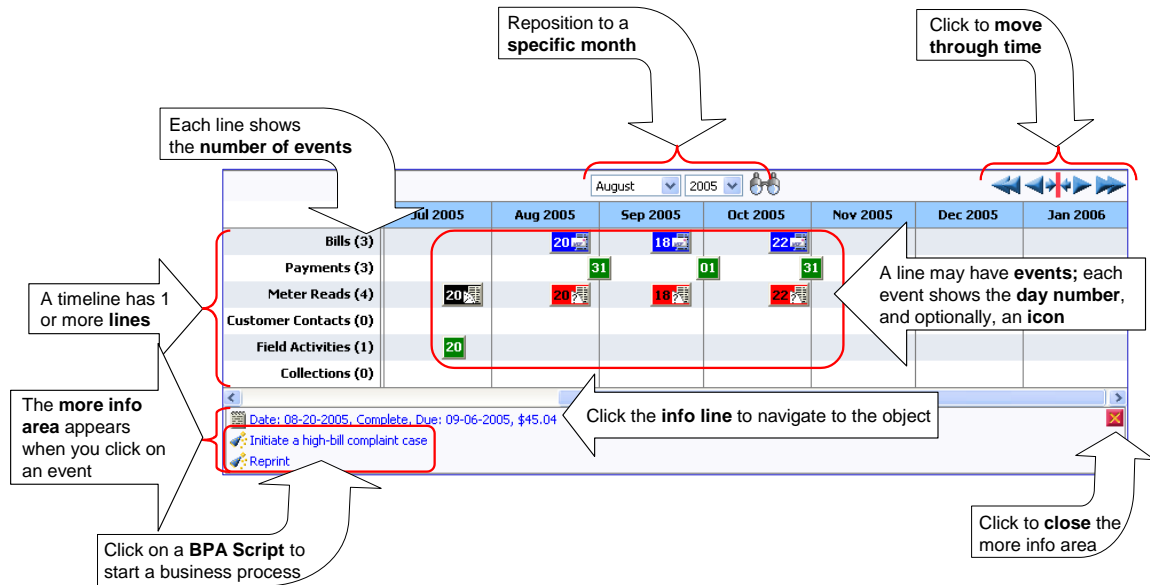
- [Timelines Zones Are Configured By Your Implementation Team](#)
- [The Anatomy Of A Timeline Zone](#)
- [You Can Move Through Time](#)
- [Timelines Can Have Many Lines](#)
- [Each Line Shows Events](#)
- [An Event's Color And Icon Can Convey Information About The Event](#)
- [An Event's Hover-Text Can Contain Additional Information](#)
- [Clicking On An Event Shows More Information In The Detail Area](#)

## Timelines Zones Are Configured By Your Implementation Team

Your implementation team controls the number and type of timeline zones you see on this page. Refer to [Configuring Timeline Zones](#) for how to add and change timeline zones.

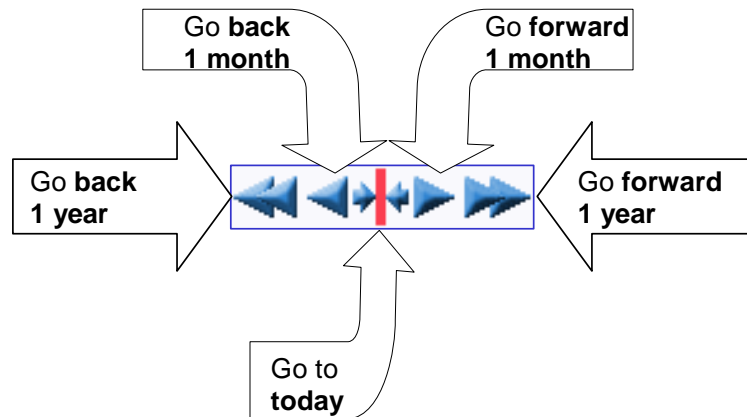
## The Anatomy Of A Timeline Zone

The following illustration highlights the various elements in a timeline zone. These elements are described in the remaining topics.



### You Can Move Through Time

You can click the controls at the top of a timeline zone to change the date-range of the zone's information:



The following points describe how to use these controls:

- To reposition the timeline to a specific date, selected the desired month and year and click the search button.
- To go back one year, click the double-left arrow.
- To go back one month, click the single-left arrow.
- To go to today, click the red line (between the arrows).
- To go forward one month, click the single-right arrow.
- To go forward one year, click the double-right arrow.

### Timelines Can Have Many Lines

A timeline zone has one or more "lines" that show when significant events have occurred. For example, you can set up a timeline zone that has two lines: one that shows when payments have been received from a taxpayer, and another that shows when bills have been sent to the taxpayer. Your implementation team controls the number and type of lines by [configuring the timeline zone](#) accordingly.

### Each Line Shows Events

Each line on a timeline may contain zero or more events where each event shows the date when the event occurs. For example, the bill line in a timeline has a separate event for every bill sent to a taxpayer. Each line's description contains the number of events on the line.

**Warning!** If a line has more events than can fit onto a timeline, the line will show the first "chunk" of events and a message will appear in the "more info area" explaining that some events have been truncated. If this happens and the truncated events are in a latter period, you can [reposition the timeline's base period](#) to show the truncated events.

### An Event's Color And Icon Can Convey Information About The Event

Your implementation team can [configure the timeline](#) so different types of events have different visual representations. For example, a timeline that shows when meter reads have occurred can use different colors and/or icons to visually differentiate between estimated and non-estimated meter reads.

### An Event's Hover-Text Can Contain Additional Information

If you hover the mouse over an event, hover text appears. Each type of event can have different hover text. For example, the hover text for a case shows significant dates in the case's lifecycle (e.g., when it was created, and when it was closed). Whereas the hover text for a meter read shows each register reading.

### Clicking On An Event Shows More Information In The Detail Area

When you click on an event, additional information appears in the zone's detail area (at the bottom of the zone). The following information may appear:

- The event's common "information string" appears. For example, if you click on a field activity event, the field activity's information string appears. This information is hyperlinked to allow for easy access to the transaction on which the object is maintained.
- Additional information may appear. For example, if you click on a bill that has been canceled and rebilled after it was initially sent to the taxpayer, information about the net result of the changes appears.
- BPA scripts that can be executed to perform business processes on the object. For example, if you click on a bill event, BPA scripts may appear that can guide you through initiating a bill dispute or canceling the bill. Note, BPA scripts are prefixed with the "wizard's hat" icon.

## Control Central - Taxpayer Information

Once [Control Central - Main](#) has a person context, you can navigate to this page to view an overview of the related persons, customer contacts, accounts, locations, and obligations linked to the account.

**User configurable.** Refer to [Each User Can Customize Which Zones Appear](#) for information about how to configure which zones appear.

## Description of Page

**Navigation hint.** The ***Go To Control Central*** option on the person context menu navigates to this tab page.

**A mouse with a roller is useful.** This page can extend vertically past the normal desktop boundary. You will find that a mouse with a “roller” will facilitate navigating through the page.

The contents of this section describe the zones that are available on this portal page.

## Contents

[Person Tree Zone](#)

[Active Account Summary Zone](#)

[Credit Allocation Detail Zone - Taxpayer Info](#)

[Timeline Zone - Taxpayer Info](#)

## Person Tree Zone

The [tree](#) in this zone shows a great deal of information including:

- All accounts linked to the person in context.
- Customer contacts linked to this person.
- The [hierarchy](#) of parents and children linked this person.
- All aliases linked to this person.

You can use this tree to both view high-level information about these objects and to transfer to the respective page in which an object is maintained.

## Active Account Summary Zone

The **Active Account Zone** lists all accounts related to the person in context. In addition, accounts linked to this person's [children](#) also appear in this zone, and if a child has children, the grandchildren's accounts are included. In fact, the system will look for accounts up to five levels deep (meaning that the great, great grandchildren's accounts will be included in this zone).

**Only accounts linked to child persons with a financial relationship are shown.** When you set up a [taxpayer hierarchy](#), you can define both subsidiaries and "key contacts" (i.e., individuals that you contact at a company). As described above, this zone will include accounts related to these subsidiaries and key contacts. However, you might not want to include the personal accounts related to the key contacts in this zone. You can control which accounts appear in this zone when you set up a hierarchy. You do this by turning on the Financial Relationship switch for those [persons](#) whose accounts should be included.

The following columns are displayed in the grid:

<b>Account Info</b>	This column contains the standard account information.
<b>Current Balance</b>	This column contains the account's current balance.
<b>Last Billed Info</b>	This column contains information about the last completed bill sent to the account.
<b>Last Contact Info</b>	This column contains information about the last customer contact associated with the account's main taxpayer.

### Credit Allocation Detail Zone - Taxpayer Info

The Credit Allocation Detail zone on the taxpayer portal shows the results of dynamic credit allocation of a person or one of its accounts or one of its obligations.

The default is to show the credit allocation detail for all obligations that support dynamic credit allocation for all accounts. Use the **Account** dropdown and click **Calculate** to restrict the information for a specific account. To further restrict by obligation, select an obligation in the dropdown and click **Calculate**.

Refer to [Credit Allocation Detail Zone](#) for information about the remainder of this zone.

### Timeline Zone - Taxpayer Info

Your implementation may have configured this page to show one or more timeline zones. Refer [Timeline Zone - Account Info](#) for a description of how to use these zones.

## Control Central - Account Tree

Once [Control Central - Main](#) has an account context, you can navigate to the **Account Tree** to see an overview of the persons, locations, and obligations linked to the account.

### Description of Page

This page is dedicated to a [tree](#) that shows the various objects linked to an account. You can use this tree to both view high-level information about these objects and to transfer to the respective page in which an object is maintained.

## Control Central - Location Tree

Once [Control Central - Main](#) has a location context, you can navigate to the **Location Tree** to see an overview of the accounts and obligations linked to the location.

### Description of Page

This page is dedicated to a [tree](#) that shows the various objects linked to a location. You can use this tree to both view high level information about these objects and to transfer to the respective page in which an object is maintained.

## Control Central - Bill/Payment Tree

Once [Control Central - Main](#) has an account context, you can navigate to the **Bill/Payment Tree** to see an overview of the financial transactions linked to the account.

### Description of Page

This page is dedicated to a [tree](#) that shows the account's bills and payments. You can use this tree to both view high-level information about these objects and to transfer to the respective page in which an object is maintained.

For [balance-forward accounts](#), bill nodes contain the balance presented on the respective bill, and pay nodes contain the amount of the respective payment. However, for [open-item accounts](#), the tree behaves differently:

- The amount on bill nodes is equal to the sum of the current charges, adjustments and corrections on the bill.
- Each bill or payment will contain an indication if all of its financial transactions are fully matched. If not, the node will become red to highlight that unmatched financial transactions exist.
- If a bill or payment node is expanded, a summary of the match status of its financial transactions is shown.

## How To Add A New Taxpayer From Control Central

There are two ways to add a new taxpayer from control central:

- Invoking the [Person](#) page from the menu bar.
- Adding a person from Control Central. You navigate as follows:
  - You start from [Control Central - Main](#). On this page, you determine if the taxpayer already exists in the system. If so, select it and you're done. If not, proceed to step 2.
  - Click the + button adjacent to the Person button to transfer to [Person - Main Information](#) page (with the name you entered already defaulted). Use the Person pages to record all relevant information about the new taxpayer. When finished, proceed to step 3.

When the person exists, you can add account(s) for it by invoking the [Account](#) page from the menu bar or from the Person context menu.

## Dashboard Portal

The [Dashboard](#) is a portal that always appears on the Oracle Enterprise Taxation Management desktop. Its zones contain tools and data that are useful regardless of the object being displayed.

**User configurable.** Refer to [Each User Can Customize Which Zones Appear](#) for information about a user can configure which zones appear.

**Global context.** The system automatically refreshes the values saved in [global context](#) with information about the object that appears in [Object Display Area](#). Various zones available in the dashboard portal have been designed to display data related to a person id, account id or location id, which are populated by the base global context algorithm [C1-GLBL-CTXT](#).

The contents of this section describe the zones that are available on this portal.

**Other zones exist.** The zones described below are unique to Oracle Enterprise Taxation Management. Please see [Core Dashboard Zones](#) for a description of other zones that can appear on the dashboard.

## Contents

[Current Context Zone](#)  
[Customer Contact Zone](#)  
[Financial Information Dashboard Zone](#)  
[Alert Dashboard Zone](#)

## Current Context Zone

The **Current Context Zone** contains basic information about the taxpayer on which you are working.

A maximum of three rows may appear:

- The first row contains a person [context menu](#) and the person's name. You can click on the person's name to transfer to the [Person - Main](#) page. Note, the person's name is formatted by a plug-in algorithm on the [installation record](#). Refer to the base package's [name format algorithm](#) for an example. If you prefer different formatting logic, your system administrator should configure the system appropriately.
- The second row contains an account [context menu](#) and the account's:
  - ID and check digit
  - The primary name of the main taxpayer
  - Taxpayer class
- The last row contains a location [context menu](#) and the location's address and location type. Note, the address information is formatted by a plug-in algorithm on the [installation record](#). Refer to the base package's [location format algorithm](#) for an example. If you prefer a different format, your system administrator should configure the system appropriately.

## Customer Contact Zone

The **Customer Contact Zone** has three purposes:

- It shows the age of the **Last** customer contact associated with the person displayed in the Current Context zone. The word **Today** is shown if the last contact was on the current date. The word **Yesterday** is shown if the last contact was on the current date - 1 day. If neither is applicable, the number of days old is shown.
- It shows the name of the person who added this contact.
- It greatly simplifies the addition of a new customer contact. To add a new contact, simply select the **Type** of customer contact, enter a brief **Comment**, and press the **Add Contact** button.

After the contact is added, the date / time of the new contact will be displayed in the **Last** area. You can use the Go To button to drill into the newly added contact where you can setup [reminders](#) or indicate the contact should [remain open until future resolution](#).



## Financial Information Dashboard Zone

The **Financial Zone** that appears on the dashboard contains the same information that is described in the [Financial Information Zone](#) on the account information portal.

## Alert Dashboard Zone

The **Alert Zone** that appears on the dashboard contains the same information that is described in the [Alert Zone](#) on the account information portal.

## Maintaining The Account Model

---

As described in Understanding The “Account Model”, the “Account Model” objects are those that form the core of your business processes. In this section, we describe the pages that maintain these objects.

### Contents

- [Maintaining Persons](#)
- [Maintaining Accounts](#)
- [Using The Account / Person Replicator](#)
- [Maintaining Tax Roles](#)
- [Maintaining Obligations](#)
- [Maintaining Locations](#)

## Maintaining Persons

On the person page, you define demographic information about your taxpayers and every other individual or business with which your company has contact. The topics in this section describe the person page.

For more information about how most persons are added, refer to [How To Add A New Taxpayer From Control Central](#). For more information about how a person is required to set up a taxpayer, refer to [Taxpayer Overview](#).

### Contents

- [Person - Main Information](#)
- [Person - Correspondence Information](#)
- [Person - Characteristics](#)
- [Person - Persons](#)
- [Person - Web Self Service](#)

### Person - Main Information

The Main page contains core person information like names, telephone numbers, and forms of identification. Open this page using **Taxpayer Information, Person, Main**.

### Description of Page

**Person Information** contains basic information about the person. These values only appear after the person exists on the database. The **ID** is a system-assigned random number that stays with a person for the life of the system.

**Formatting is performed by a plug-in.** The format of **Person Information** is controlled by a plug-in algorithm on the [installation record](#). Refer to the base package's [person information algorithm](#) for an example. If you prefer different formatting logic, your system administrator should configure the system appropriately.

The following information may be recorded about a person:

Define the **Person Type** for this person. The person type controls some behavior for the person. It includes a "person/business" indicator that controls the primary name validation. Your implementation's person types may also be configured to define other behavior for persons of this type such as the default primary ID type.

**Person Names** are used by [Control Central](#) to look for accounts and persons. In addition, a person's primary name is the addressee on the person's correspondence unless overridden by the Override Mailing Name (maintained on the **Misc** tab). The following fields display:

- Use **Name Type** to indicate if the name is an *Alias*, *Alternate Representation*, *Doing Business As*, *Legal*, or *Primary* name. Note, for new persons, a value of *Primary* is defaulted.

**Note.** The values for the name type field are customizable using the Lookup table. This field name is NAME\_TYPE\_FLG.

- Use **Person Name** to define the person's name. Note well, the name is case sensitive.

**Alternate representations of a person's name.** You would use an *Alternate Representation* for a person's name when you have an alternate ways to define the person's primary name. Alternate representations are typically used in countries that use multiple character sets (e.g., the *Primary* name is entered in Chinese, the *Alternate Representation* is entered in English). When a person has an alternate name, both the main and alternate names can be used to search for a person. The *Alternate Representation Name Type* only appears if you have enabled alternate names on the installation record. Refer to the description of the Alternate Representation field under [Installation - Main](#) for more information.

**Validation is performed by a plug-in.** The validation that is applied to **Person Name** (e.g., a comma separating the last and first name - *Smith,Patricia*) is controlled by a plug-in algorithm on the [installation record](#). Refer to the base package's [name validation algorithm](#) for an example. If you prefer different validation logic, your system administrator should configure the system appropriately.

**Person Phone** numbers are used by [Control Central](#) to look for accounts and persons. The following fields display:

- **Phone Type** indicates the type of phone number, e.g., Home, Mobile, Business, ...
- Use **Phone Number** to define the telephone number. Enter the telephone number in the format described by the **Phone Format**.

**Formatting is performed by a plug-in.** The format that is applied to a **Phone Number** is controlled by the algorithm that is plugged in on the respective [Phone Type](#). If you prefer a different format, your system administrator should configure this algorithm appropriately.

- Enter the **Extension**, if any, of the telephone number.

A **Person's ID's** have several uses:

- They are used by [Control Central](#) when you look for a taxpayer / location based on their ID.
- They are used to highlight potential duplicate persons.
- [Control Central](#) displays a person's **primary** identification in the search results area to help a user identify the taxpayer when multiple taxpayers match the search criteria.

**Person ID may be required or optional.** The person ID usage flag on the [installation record](#) indicates whether at least one id for a person is **required** or **optional**.

The following fields are used to define a person's ID(s).

- Turn on **Primary ID** for the piece of **ID** that is the primary means of identifying the taxpayer.
- Indicate the **ID Type**. The **ID Type** defaults from the [Installation Record](#) based on the **Person Type's** person/business indicator.
- Enter the identification number in the adjacent fields. Please note that if the **ID Number** should be formatted (e.g., dashes in an American social security number), you do not have to enter the dashes. Rather, you can enter the information as a contiguous value and the system will format this for you. The format is shown in the adjacent **Identifier Format** column.

**Formatting is performed by a plug-in.** The format that is applied to an **ID Number** (e.g., dashes in an American social security number) is controlled by the algorithm that is plugged in on the respective [ID Type](#). If you prefer a different format, your system administrator should configure this algorithm appropriately.

## Person - Correspondence Information

This page contains information that may be used to address [bills](#) and [letters](#). Use **Taxpayer Information, Person, Correspondence Info** to open this page.

### Description of Page

If the person does not want their primary name (defined on the Main page) used on [bills](#) and [letters](#), specify the desired name in **Override Mailing Name 1, 2, and 3**.

Specify the **Override Mailing Address** fields if the person wants their bills and letters sent to an address specific for this person. In addition, if the person's account doesn't already indicate that the person's mailing address should be used, you must update this person's account(s). This information resides in the Address Source field on [Account - Person](#).

**Address isn't everything.** In addition to defining the person's **Override Mailing Address**, there may be additional tasks you must perform in order to route information to this address. Refer to the following links for more information: [where are bills sent](#) and [where are letters sent](#).

If you enter an **Override Mailing Address**:

- The **Country** defaults from your [installation options](#).
- The address constituents may differ depending on the **Country**. Refer to [Defining Countries](#) for more information on the address constituents.
- If you have set up [postal defaults](#), the system will default the address constituents when you tab out of the postal code.

Specify the taxpayer's **Email Address** if you communicate with the taxpayer via Email. In addition to defining the person's **Email Address**, your correspondence routing software must support sending the information via Email. Refer to the following links for more information: [where are bills sent](#) and [where are letters sent](#).

Define the **Language** in which the person prefers their bills and correspondence printed.

**Default note.** The person's language defaults from [Installation Options - Person](#).

Refer to [Taxpayer Language](#) for more information on options for supporting multiple languages for your taxpayers.

The **Miscellaneous Addresses** scroll to define additional addresses for the person.

- Select the **Address Type**. The base product supports a value of **Seasonal**. See below for additional detail about seasonal addresses. Your implementation may introduce additional values for this field.

**Note.** Define values for this field using the Lookup table. This field name is ADDR\_TYPE\_FLG.

- The **Status** of the address must be **Active**. You can set the status to **Inactive** if you want a seasonal address ignored (alternately, you can just remove the seasonal address).
- The **Country** defaults from your [installation options](#). The address constituents may differ depending on the **Country**. Refer to [Defining Countries](#) for more information on the address constituents.
- If you have set up [postal defaults](#), the system will default the address constituents when you tab out of the postal code.

The **Seasonal** address type allows your taxpayer to define one or more alternate addresses to use during predefined periods each year. For example, the taxpayer may want their correspondence sent to their vacation cottage during the summer. Please be aware of the following:

- Seasonal addresses will only be used if the taxpayer's bills and / or letters are routed via the postal service. For example, if you route bills to the taxpayer via Email, the seasonal address will never be used to route bills to the taxpayer. Refer to the following links for more information: [where are bills sent](#) and [where are letters sent](#).

- You don't have to specify a seasonal address for every part of the year. For example, if the taxpayer wants their bills sent to their main address except during the summer, you need only enter a seasonal address for the summer.
- You can enter a seasonal address with or without an **Override Mailing Address**. If an **Override Mailing Address** is not specified, the person's correspondence will be addressed as per the instructions on the person's account(s). These instructions reside in the Address Source field on [Account - Person](#).
- You can enter multiple seasonal addresses if the taxpayer so desires.
- The **Seasonal** period is defined in the two **Season** fields. The first field contains the day and month when the season starts; the second field contains the day and month when the season ends. The day and month should be entered in the format defined in your [display profile](#).

## Person - Characteristics

The characteristics page contains information that describes miscellaneous information about the person. Use **Taxpayer Information, Person, Characteristics** to open this page.

### Description of Page

**Note.** You can only choose characteristic types defined as permissible on the person record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

The following fields display:

<b>Effective Date</b>	Define the date on which the characteristic becomes effective. The effective date defaults from the <a href="#">Installation Record</a> .
<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

## Person - Persons

This page is used to define this person's relationship with other persons. For example, if the person being maintained is a conglomerate, you can define its subsidiaries on this page. Refer to [How To Set Up Taxpayer Hierarchies](#) for more information about hierarchies.

**You're defining "child relationships".** It's important to understand that the persons being defined on this page's grid should be thought of as "children" of the person on the top of the page.

**Children may have children.** It's possible for one of the children to have children itself (for example, if you have a situation where a subsidiary of a conglomerate itself has subsidiaries). To define a child's children, simply display the child person on this page and then define its children.

**Equal relationships.** It's possible to link persons where no hierarchy is implied by the relationship (e.g., spouses). There are two ways to do this: 1) you can nominate one spouse as the "parent" and the other as the "child" or 2) you can define the spousal relationship for both persons (i.e., you would define Robert Chopin as the husband of Jeanette Chopin, and Jeanette Chopin as the wife of Robert Chopin). If you choose the latter approach, a recursive relationship will exist.

**Warning!** If your organization enters multiple levels of person, we want to point out that we do not prevent recursive relationships. This means that you could set up a nonsensical situation where person 2 is a child of person 1 and person 1 is a child of person 2.

Use **Taxpayer Information, Person, Persons** to open this page.

### Description of Page

The grid contains the "children" associated with the "parent" person who is defined at the top of the page. The following fields display:

<b>Child Person</b>	This is the unique identifier of the "child" person. This person's main name appears adjacent.
<b>Child Information</b>	An informational message appears to highlight if the child person itself has children. You can click the corresponding go to button to view the child's children.
<b>Relationship Type</b>	Indicate the type of relationship between the parent and the child person.
<b>Start Date</b>	Indicate the date on which this relationship began. This field defaults from the <a href="#">Installation Record</a> .
<b>End Date</b>	If the relationship expires, indicate the date the relationship stops.
<b>Financial Relationship</b>	Turn on this switch if the child person has account(s) and information about these account(s) should be displayed when the parent person's hierarchy is displayed in the <a href="#">Active Account Summary Zone</a> .

### Person - Web Self Service

This page is used to define information for this person to access account information via your web self-service application.

**Configuring Web Self Service.** The system provides sample web self-service functionality. This functionality may be used as a basis for implementing a full web application for your organization that is integrated with other web services you offer.

Use **Taxpayer Information, Person, Web Self Service** to open this page.

### Description of Page

The information on this page is entered by the taxpayer when registering for the web self service application.

**Web User ID.** The sample web self service functionality provided by the system defines the taxpayer's web user ID using an entry in the person ID collection defined on the [main](#) tab.

The **Web Self Service Password** chosen by this person through the web application is stored encrypted and cannot be viewed. Taxpayers should change their password via the web application.

The **Web Self Service Password Hint** and **Web Self Service Password Answer** are defined by the taxpayer when registering and are used when the taxpayer has forgotten the password.

**Note.** The values for the password hint field are customizable using the Lookup table. This field name is WEB\_PWD\_HINT\_FLG.

The **Web Self Service Receive Marketing Info Flag** indicates whether or not the taxpayer has chosen to receive marketing information. The possible values are *Receives Marketing Info* and *Doesn't Receive Marketing Info*. This information may be used by your web application to optionally send marketing information to this taxpayer via email.

## Maintaining Accounts

The account record contains information that controls billing and collections processing. You should only need to use this page if you need to fine-tune the information that the system has set up by default.

The topics in this section describe the pages on which account related information is maintained.

### Contents

- [Account - Main Information](#)
- [Account - Auto Pay](#)
- [Account - Person Information](#)
- [Account - Bill Messages](#)
- [Account - Collections](#)
- [Account - Characteristics](#)
- [Account - Alerts](#)

### Account - Main Information

The Main page contains core account information. Open this page using **Taxpayer Information, Account**.

#### Description of Page

The primary name of the **Account's** main taxpayer and the **Account ID** are displayed on every tab in this page. These values only appear after the account exists on the database. The **Account ID** is a system-assigned random number that stays with an account for life.

**Formatting may be performed by a plug-in.** The basic information about an account that appears at the top of this page (and on many other pages in the system) may be formatted by a plug-in algorithm on the [installations record](#). Refer to the base package's [account information algorithm](#) for an example. If you prefer different formatting logic, your system administrator should configure the system appropriately.

A “check digit” is displayed adjacent to the account ID. This is for information purposes only, and is not needed to operate the system. The following points describe how the check digit is calculated for an Account ID equal to **0011883422**.

- Calculate the sum of the first and every alternate digit in the account id.  $A = 14 = 0+1+8+3+2$
- Calculate the sum of the second and every alternate digit in the Account ID and multiply by 2. For example,  $B = 30 = (0 + 1 + 8 + 4 + 2) * 2$
- Add A and B. For example,  $C = 44 = 14 + 30$
- Count the number of digits used to calculate B that are greater than 4. For example,  $D = 1$  since only 8 is greater than 4.
- Multiply D by 9. For example,  $E = 9 = 1 \times 9$
- Subtract E from C. For example,  $F = 35 = 44 - 9$
- Subtract the units position of F from 10 to find the check digit. For example, check digit =  $5 = 10 - 5$  (5 is subtracted from 10 because  $F = 35$  and there the units position is 5).

**Technical note.** The above is calculated in the common routine called **CIPACDN**.

**Set Up Date** is the date the account was initially set up. This is purely informational.

**Currency Code** defines the currency in which the account's financial transactions are expressed. All rates and payments associated with this account must be denominated in this currency.

**Default note.** The currency defaults from the [Installation Record](#) and may be overridden here.

**Account Type** plays a part in:

- The account's default collection class and when the overdue monitor reviews an account. Refer to [How Does The Overdue Monitor Work](#) for more information about how and when an account's debt is reviewed.
- And several other functions. Refer to and [Setting Up Account Types](#) for more information.

**Default note.** The account type defaults from [Installation Options - Account](#) (Person Account Type) and may be overridden at will.

**Division** defines the jurisdiction that governs this account. You may only select **Divisions** associated with the account's [account type](#). This field is updated behind the scenes every time an obligation is activated (the system uses the **division** associated with the obligation's obligation type). If you have assigned a division and do not want the system to change it when an obligation is activated, turn on **Protect division**.



**Division governs many functions.** An account's division impacts its subsequent bill due dates, collections review dates, the roles assigned to To Do entries, and the calendar of workdays. Refer [Setting Up Account Types](#) for more information.

**Access Group** controls which users are allowed to view and update this account's information (including bills, obligations, payments, locations, ...). The system defaults this value from the user's [default access group](#). Refer to [The Big Picture of Row Security](#) for a complete description of how account security is implemented in the system.

The optional Account **Management Group** controls the roles assigned to [To Do entries](#) associated with an account. Refer to [Setting Up Account Management Groups](#) for more information.

Enter a **Comment** to define unusual information about the account. If this field is populated, an alert will highlight such in the [Alert Zone](#).

**Bill Cycle** controls when a bill is produced for an account. The route the system takes to do this update is as follows:

If you have assigned a bill cycle and do not want the system to change the bill cycle when an obligation is activated, turn on **Protect Bill Cycle**.

**Mailing Location ID** defines the address on bills for persons who have their bill sent to the account's mailing location. This field may be updated behind the scenes when an obligation is activated. The word "may" is used because the system will only update an account's mailing location if there are no **active** obligations currently linked to the account. Refer to [Account - Person Information](#) for more information about how to define where a person has their correspondence and bills sent.

If you do not want the mailing location to change when an obligation is activated, turn on **Protect Mailing Location**.

If you want to stall billing until after some future date, enter the date in **Bill After**.

If user should review the account's printed bills before they are sent to the taxpayer, enter the user ID of the individual who reviews the bill in **Bill Print Intercept**.

## Account - Auto Pay

The Auto Pay page is used when an account pays their bills automatically (e.g., by direct debit or credit card). The Auto Pay page can also be used when a taxpayer chooses to receive their refunds via a direct deposit.

Refer to [How And When Are Automatic Payments Created](#) for more information.

Open **Taxpayer Information, Account** and navigate to the **Auto Pay** tab to maintain this information.

### Description of Page

The **Account Auto Pay** scroll defines the bank account / credit card from which the taxpayer's automatic payments are debited. Multiple auto-pay options may exist if the taxpayer changes auto-pay options over time. The following fields display:

- **Start Date** is the date on which the automatic payment information comes into affect. The system creates an automatic payment for any bill produced for this account with a due date on/after this date and on/before the **End Date**. If **End Date** is not specified, this means the automatic payment option applies indefinitely. You need only specify an **End Date** if the taxpayer wants to stop paying automatically.
- **Auto Pay ID** is the unique, system-assigned identifier of the auto-pay record. This value is assigned after the information is saved on the database and may not be modified.
- Use **Auto Pay Source Code** to define the source of the funds used to satisfy the automatic payment request. For example, if a taxpayer indicates that they want to use their checking account to pay their bill, you would specify the **Auto Pay Source Code** associated with their bank. The source's description and external source ID (e.g., bank routing number) are displayed adjacent.
- Use **Auto Pay Method** to specify whether you want the system to process automatic payments as **Direct Debit** or **Payment Advice**. This field is visible only if feature configuration is set up to support payment advice functionality. Refer to [Payment Advices](#) for more information on the payment advice functionality.
- Use **External Account ID** to define the taxpayer's bank account / credit card number.
- You will be required to define an **Expires On** date if the **Auto Pay Source Code** references a tender type that requires an expiration date (e.g., if the Auto Pay Source is a credit card company).
- Enter the **Name** of the taxpayer as it appears in the financial institution's system. This name is routed to the financial institution.

**Default note.** The **Name** will default to the primary name of the main taxpayer linked to the account after you enter a **Start Date**. This defaulting is only possible for accounts that exist on the database.

- In some locales, taxpayers can define a **Maximum Withdrawal Amount** to limit the amount of money that is automatically debited from their account. Refer to [How To Implement Maximum Withdrawal Amounts](#) for more information.
- Use **Comments** to describe anything interesting / unusual about the automatic payment request.

## Account - Person Information

The Account Person page contains information about the person(s) linked to the account.

**Note.** Both taxpayers and non-taxpayers (e.g. the accounting service that handles the bills of a commercial taxpayer) can be linked to an account.

If you need to link multiple persons to an account (e.g., because you need to send a duplicate of the account's bill to someone other than the main taxpayer) or change information about a person on an account, open **Taxpayer Information, Account** and navigate to the **Persons** tab.

## Description of Page

The **Account Persons** scroll contains one entry for every person linked to the account. The following fields display:

<b>Person ID</b>	The unique identifier of the person linked to the account. The person's name is displayed adjacent.
<b>Primary Taxpayer</b>	<p>Turn on this switch if the person is the main taxpayer on the account. Only one person on an account may be designated as the main taxpayer.</p> <p>The significance of the main taxpayer is that its name will appear adjacent to the account ID throughout the system.</p>
<b>Financially Responsible</b>	<p>Turn on this switch if the person is financially responsible for the account's debt.</p> <p>This switch is on and gray if the person is the <b>Main Taxpayer</b> because the main taxpayer is always financially responsible. If multiple persons are linked to the account, you use this switch to indicate which ones are financially responsible versus those that are linked for other purposes.</p>
<b>Third Party Guarantor</b>	Turn on this switch if the person is a 3 <sup>rd</sup> party guarantor of the account's debt. This switch is off and gray if the person is the <b>Primary Taxpayer</b> .
<b>Relationship Type</b>	Define the relationship between the person and the account. Refer to <a href="#">Setting Up Account Relationship Codes</a> for information about setting up relationship types.
<b>Web Access</b>	Indicate whether or not a person linked to this account is <b>Allowed</b> or <b>Not Allowed</b> to view details of this account via your web self service application.

**Default note.** This flag is defaulted to **Allowed**.

<b>Prefix/Suffix and Pfx/Sfx Name</b>	Use these fields if you need to append additional information to a taxpayer's name when correspondence or bills are sent to this person. For example, if the account has declared bankruptcy you might attach "Debtor In Possession" to the person's name. Use <b>Prefix/Suffix</b> to indicate if the <b>Pfx/Sfx Name</b> should be appended to the front or the back of the taxpayer's name.
---------------------------------------	--

**Receives Copy of Bill**

Turn on this switch if the person receives a copy of the account's bills. Turning off this switch grays the remaining fields. This switch is on and gray if the person is the **Primary Taxpayer** because the main taxpayer always receives a copy of the bill.

Refer to [The Source Of Bill Routing Information](#) for more information about how the remaining information is used to format the address of a bill.

**Bill Route Type**

Indicate how the bill is sent to the taxpayer. This field's value defaults from the [Installation Record](#). If the **Bill Route Type** you select indicates that bills are routed via Fax, the person's fax number is displayed adjacent (the system knows which of a [person's phone numbers](#) is a fax number by the phone type). If the **Bill Route Type** you select indicates that bills are routed via Email, the person's [Email address](#) is displayed adjacent. If the **Bill Route Type** you select indicates that bills are routed via the Postal service, you must choose an appropriate **Address Source** to define which address should be used. Refer to [Setting Up Bill Route Types](#) for more information about bill route types.

**Bill Format**

Indicate if the taxpayer should receive a **Detailed** or a **Summary** bill.

**Note.** The values for this field are customizable using the Lookup table. The values need to match the formats supported by your bill print software. This field name is BILL\_FORMAT\_FLG.

**Number of Bill Copies**

Indicate how many copies of the bill the person receives.

**Taxpayer PO ID**

Indicate the purchase order ID the taxpayer wants printed on their copy of the bill.

**Receives Notification**

Turn on this switch if the person receives letters (i.e., notifications) initiated by collection, severance, write-off, overdue and cut events. These events will send letters to the main taxpayer and any other person linked to the account that **Receives Notifications**. This switch is on and gray if the person is the **Main Taxpayer** because the main taxpayer always receives notifications. Refer to [Printing Letters](#) for more information about how letters are produced by the system.

**Address Source**

If the **Bill Route Type** you selected indicates that bills are routed via the postal service, you must choose the appropriate **Address Source**.

Choose **Mailing Location on Account** if bills should be sent to the address associated with the Mailing Location on the first page. This address is displayed adjacent.

Choose **Person** if bills should be sent to the [person's mailing address](#). This address is displayed adjacent.

Choose **Account Override** if bills should be sent to an override address specified below. Typically, you would only choose this option if the person has multiple accounts and each account's bills should be sent to a different address. If you choose this option, the bottom of this page will become populated with fields that should be used to specify this override address.

#### Override address constituents

If you selected an **Address Source** of **Account Override**, you must enter the address to which bills will be sent in the address constituents. The number and type of address constituents is based on the [Country](#). These fields will be invisible for other **Address Sources**.

Note, the **Country** defaults from [Installation Options - System](#). The City, County and State default based on the **Country** and **Postal Code** if a [Postal Default](#) exists for the postal code.

### Account - Bill Messages

The Account Message page contains information about the message(s) to appear on an account's bills. Both one-time and permanent messages may be defined. This information is used when the system assembles messages to appear on an account's bill. Refer to [The Source of Bill Messages](#) for more information about bill messages.

Open **Taxpayer Information, Account** and navigate to the **Bill Messages** tab to access this page.

#### Description of Page

The following fields are displayed:

##### Bill Message

The message code and description of the message that appears on the taxpayer's bill.

##### Bill Message Type

Use **Temporary** to indicate the message should only be linked to the next bill produced for the account. Use **Permanent** if the message should appear on every bill. Note, a value of **Temporary** defaults.

### Account - Collections

The collections page contains the compliance rating transactions that impact an account's compliance rating. To view or modify these transactions, open **Taxpayer Information, Account** and navigate to the **Collections** tab.

## Description of Page

**Collection Class** controls how the account's debt is compared against collection criteria to determine if an overdue process should be started. Refer to [How Does The Overdue Monitor Work](#) for more information about how and when an account's debt is reviewed.

**Default note.** The **Collection Class** defaults from the account's Account Type when the account is first added. It may be overridden at will.

If you need to prevent an account from being reviewed by the overdue monitor, use **Postpone Compliance Review Until** to define the future date when these processes can again review the account's debt.

**Last Compliance Review Date** is the date when the account's debt was last reviewed by the overdue monitor ([C1-ADMOV](#)).

The account's **Current Compliance Rating** is displayed. For an explanation of how this number is derived, see [How is an account's compliance rating calculated?](#)

The **Compliance Rating History** scroll contains one entry for each compliance rating history record associated with the account. The following fields display:

**Start Date** This is the first date the compliance rating transaction affects the account's compliance rating.

**End Date** This is the last date the compliance rating transaction affects the account's compliance rating.

The **Created On** message is formatted using an algorithm on the [Installation Options](#).

**Affect Compliance Rating By** This is the effect of the compliance rating transaction on the account's compliance score. This should be a *negative* number because the *lower* the score the worse the compliance rating.

An account's compliance rating is equal to the start compliance rating amount defined on the [installation record](#) plus the sum of compliance rating demerits that are currently in effect. When an account's compliance rating is less than the compliance rating threshold defined on the [installation record](#), the account's compliance rating is displayed as an alert in the [Alert Zone](#).

**Comments** Enter comments to clarify the reason(s) for the creation of the transaction.

## Contents

[How are compliance rating transactions created?](#)

[How is compliance rating calculated?](#)

[What impact does compliance rating have in the system?](#)

How are compliance rating transactions created?

Compliance rating transactions may be created as follows:

- If a payment tender is canceled with a cancellation reason that indicates the cancellation was due to non-sufficient funds, a compliance rating transaction is created. The number of compliance rating is defined on the [payment cancellation reason code](#).
- A user may create compliance rating transactions at their discretion.
- Algorithms may add compliance rating transactions. The system provides a sample overdue event algorithm that adds compliance rating transactions. Refer to [C1-OE-CR-RT](#) for more information.

**Created By Flag.** The compliance rating transaction includes a **Created By** flag that is customizable using the Lookup table. This field name is CR\_RAT\_CRE\_BY\_FLG. If your implementation uses algorithms to create compliance rating history transactions, you may consider adding appropriate values to the Created By flag. Also note that the Created By flag includes the value **Other**, which may be used by your implementation specific algorithms if you do not want to add a new custom value.

#### How is compliance rating calculated?

An account's compliance rating is calculated by summing the Affect on Compliance Rating from compliance rating transactions effective on the current date. This value is added to the perfect compliance rating defined on the installation record. The result is the taxpayer's current compliance rating.

#### What impact does compliance rating have in the system?

The following points describe the impact of an account's compliance rating:

- An account's current compliance rating is displayed in the Control Central Alert area when it's less than the threshold compliance rating defined on the installation record.
- An account's compliance rating may affect how the account's debt is collected. We used the word "may" because an account's compliance rating will only affect collection criteria if you set up your overdue processes to do this.

### Account - Characteristics

The characteristics page contains information that describes miscellaneous information about the account. Use **Taxpayer Information, Account** and navigate to the **Characteristics** tab to open this page.

#### Description of Page

**Note.** You can only choose characteristic types defined as permissible on the account record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

The following fields display:

#### Effective Date

Indicate the date on which the characteristic value becomes effective. Note, the effective date defaults to the account's setup date (defined on the **Main** page).

#### Characteristic Type

Indicate the type of characteristic.



**Characteristic Value**

Indicate the value of the characteristic.

**Account - Alerts**

The alerts page contains information that describes various alerts assigned to the account. Use **Taxpayer Information, Account** and navigate to the **Alerts** tab to open this page.

Alerts assigned to an account appear in the [Alert Zone](#). They are used to bring important information to the attention of any user who looks at the account. Up to 60 alerts can display in the Alert Zone.

**Description of Page**

The following fields display:

**Alert Type**

Indicate the type of alert to show on Control Central.

**Start Date**

Indicate the date on which the alert starts showing in Control Central. The current date defaults.

**End Date**

Indicate the date on which the alert stops showing in Control Central. You can leave this field blank if the alert should be effective indefinitely. The date defaults to the **Start Date** plus the **Alert Type's** alert period.

**Using The Account / Person Replicator**

There are two different ways to use this transaction:

- You can make many copies of a given Person / Account combination. This may be required if your company works with 3<sup>rd</sup> party agencies who enroll taxpayers on its behalf. In this situation, the 3<sup>rd</sup> party agency may receive blocks of pre-allocated account numbers to be used when they enroll a taxpayer. When a taxpayer signs up, the person information can be updated with information about the newly enrolled taxpayer.

Perform the following steps if you need to create many copies of a given person and account:

- [Add a new person and account](#) if you don't already have a taxpayer to serve as your "template".
- Use the account / person replicator to create copies of the person and the account as described below (and make sure to use the **Replicate Acct and Person** option).
- You can create many Accounts for a single Person. This may be required if you have a taxpayer who wants many separate accounts rather than a single consolidated account.

Perform the following steps if you need to create many copies of a given account for a specific person:

- Select the account that serves as the template account for the person. Note, this account must be linked to the person who wants the many accounts.
- Use the account / person replicator to create copies of the account as described below (and make sure to use the **Replicate Account Only** option).

Use **Taxpayer Information, Account/Person Replicator** to open this page.

**Description of Page**



Choose the **Account** that serves as the template account.

**Recommendation.** Carefully verify that the template account you have chosen is correct before you save the replicated accounts. After you save the replicated accounts, any corrections could prove time consuming.

At the top of the page, indicate the **Number** of accounts and persons **to Replicate**.

Indicate the **Main Person Name Base**, which will be used as the main person name for all the new persons to be created. This field will be protected if you choose a **Replicate Type** of **Replicate Account Only** (as no persons are created).

Use **Replicate Type** to define if you want to create multiple copies of an account for a given person or if you want to create many new person / account combinations. See the description that appears above for more information about this field.

After entering the above, click the **Replicate** button to generate your new accounts and persons. The grid displays informational messages describing what will happen when you click save. For example, it will verify how many accounts will be created, the person whose record will be copied, the name that will be used on all the new persons, etc..

If everything looks clean, click save.

## Contents

[Information Replicated for New Persons](#)

[Information Replicated for New Accounts](#)

## Information Replicated for New Persons

Except for the main name, which you indicate in the **Main Person Name** field, all information for the main person linked to the account will be copied to the new persons.

**Warning!** This copy includes the person's identification number. This will result in multiple persons with the same identification number. You should ensure that the identification number for your template person is a temporary number. When you sign up your actual taxpayer, be sure to update the identification number to a valid number for the person.

## Information Replicated for New Accounts

Although most information for an account will be copied, not all information will be. The following points describe the information that is NOT copied on newly replicated accounts:

- Mailing location will be set to blank
- Setup Date is set to the current date
- Compliance Review Date and Postpone Compliance Review Until Date will be set to blank
- Auto Pay information will not be copied
- Compliance Rating History will not be copied

**Note** that all miscellaneous persons linked to the account (i.e., those who are not flagged as Main Taxpayers) will be linked to the new accounts. With this functionality, for example, you can create a person for a third party enroller and link this enroller to the template account. This enroller will then be linked to all the new accounts.

## Maintaining Tax Roles

Use the Tax Role transaction to view and maintain tax roles. Navigate using **Main Menu, Taxpayer Information, Tax Role**.

### Contents

- [Tax Role Query](#)
- [Tax Role Portal](#)

### Tax Role Query

Use the [query portal](#) to search for an existing tax role. Once a tax role is selected, you are brought to the maintenance portal to view and maintain the selected record.

### Tax Role Portal

This portal appears when a tax role has been selected from the Tax Role Query portal.

The topics in this section describe the base-package zones that appear on this portal.

### Contents

- [Tax Role Actions](#)
- [Tax Role](#)
- [Related Obligations](#)

#### Tax Role Actions

This is a standard [actions zone](#).

#### Tax Role

The Tax Role zone contains display-only information about the tax role.

Please see the zone's help text for information about this zone's fields.

#### Related Obligations

This zone lists obligation currently linked to the broadcast tax role.

## Maintaining Obligations

The topics in this section describe the pages on which obligation related information is maintained.

**Warning!** Take special care when adding a new obligation to specify the appropriate division and obligation type as it will affect how the taxpayer is billed, what rate can be specified, how overdue debt is collected, and much more. After the obligation is activated, you may not change its obligation type.

## Contents

- [Obligation - Main Information](#)
- [Obligation - Rate Info](#)
- [Obligation - Chars, Qty & Rec. Charges](#)
- [Obligation - Miscellaneous](#)

## Obligation - Main Information

The Main page contains basic obligation information. Open **Taxpayer Information, Obligation** to maintain this information.

### Description of Page

**Obligation Info** and **Obligation ID** are displayed on every page. These values only appear after the obligation exists on the database. The **ID** is a system assigned random number that stays with an obligation for life. The **Obligation Info** is a concatenation of important details about the obligation and its account.

**Formatting may be performed by a plug-in.** The contents of **Obligation Info** may be formatted by a plug-in algorithm on Obligation Type. Refer to the base package's [C1-SAT-INFO](#) for an example. If such an algorithm is not plugged-in on the obligation type, the system looks for a corresponding algorithm on the [installation record](#). Refer to the base package's [C1-SAI-INFO](#) for an example. If you prefer different formatting logic, your system administrator should configure the system appropriately.

**Obligation Status** defines the state of the obligation. The system typically changes the state of an obligation behind-the-scenes. If you can't wait for the system, use one of the adjacent buttons:

- Click the **Activate** button to activate a **Pending Start** obligation.
- Click the **Cancel** button to cancel a **Pending Start** or **Active, Pending Stop** or **Stopped** obligation.
- Click the **Stop** button to stop a **Pending Stop** obligation.
- Click the **Close** button to close a **Stopped** or **Reactivated** obligation.

Enter the **Account ID** that is financially responsible for the obligation. If you change an obligation's **Account ID**, you are effectively transferring this obligation (and its debt) to the new account.

Indicate the **Division** and **Obligation type**. These fields are very important as they control numerous aspects of the obligation's behavior. These fields are gray when the status is not **Pending Start**.

For more information about what the obligation type controls, refer to [Setting Up Obligation Types](#).

If the account is populated and the obligation type's tax type indicates a [tax role](#) is required, enter the **Tax Role** for this obligation.

If the [obligation type](#) indicates that a filing period required, define the **Filing Period** for the obligation.

The **Start Date** defines when the financial relationship begins. The **End Date** defines when the financial relationship terminates. If the end date is blank, the obligation has not yet been terminated.

Enter an **Override Filing Due Date** if the taxpayer has requested an extension to the filing [due date](#) for this obligation's filing period.

Enter an **Override Payment Due Date** if the taxpayer has requested an extension to the payment [due date](#) for this obligation's filing period.

Use **Maximum Bill Threshold** if you want the system to generate a bill error when a bill segment is produced [in batch](#) that exceeds a given value. These bill errors will appear on the standard billing queries and To Do lists. If, after reviewing the high value bill segment, a user truly intends to send the bill out, they should regenerate the bill. Refer to [How To Correct A Bill Segment That's In Error](#) for more information.

**Default note.** The value of **Maximum Bill Threshold** defaults from the obligation's obligation type.

If you've set up the system to use start options, you can have the system default many fields by copying the terms from a start option when service is started. The **Start Option** field contains the last start option whose field values were copied to the Obligation. If you need to change an Obligation's terms by applying a different start option, you can. Refer to [Changing A Start Option](#) for a description of how to do this.

The **Char Location ID** is a unique identifier for the location, or physical address. For obligation types such as property tax, certain characteristics of the location will be needed in order to properly assess the obligation amount. Other obligation types such as sales tax will require a valid location for which to send correspondence to.

Your conversion process will fill in **Old Account ID** for obligations that were converted from your legacy system. The [payment upload process](#) uses this field to locate the accounts for payments that reference a legacy account number.

If the Obligation requires a total amount to bill, the amount is displayed. The label that appears for this field is defined on the Obligation type (on the Billing tab). We recommend that the system be set up so that this label is very specific.

The **Pay Plan Autopay** flag determines if the pay plan scheduled payments are excluded from automatic payment or included by automatic payment. If the account is not set up for automatic payment for the period that covers the pay plan, this flag cannot be set. This field is only visible if the obligation has a special role of **Pay Plan**.

The **Recommendation Rule** displays the description of the pay plan recommendation rule used for this obligation. This field is only visible if the obligation has a special role of **Pay Plan**.

This bottom of this page contains a [tree](#) that shows the various objects linked to the obligation. You can use this tree to both view high-level information about these objects and to transfer to the respective page in which an object is maintained.

The topics that follow describe miscellaneous features on this page.

### Changing A Start Option

If the obligation is in the **active** or **pending stop** states and if the obligation type uses start options, a button appears on obligation - Main called **Apply New Start Option**.

Click on the button to bring up the Apply New Start Option Confirmation popup. The fields in this dialog allow you to define a **Start Option** whose terms should be copied to the obligation as of a given effective date (**Apply Start Option on**). Because most terms on an obligation are effective-dated, it's important that you understand the following ramifications of applying a start option (note, we use the word "terms" to generically reference any of the fields that can be defaulted from a start option):

- Terms are reused if **Start Option** contains the same value as defined in the obligation. For example,
  - If the **Start Option** references the same rate as defined in the obligation, a new rate record will NOT be created.
  - If the **Start Option** references a contract rider that's currently defined in the obligation, a new rider will NOT be created.
- Terms that support explicit expiration (i.e., those with an end date) are expired if they are not referenced on the new **Start Option**. For example, if the obligation has a rider that is not referenced on the new **Start Option**, it will be expired.
- Terms that support implicit expiration (i.e., those without an end date) are left unchanged if they are not referenced on the new **Start Option**. For example, an obligation characteristic that isn't referenced on the new **Start Option** will NOT be modified when a new start option is applied.

**Only start options related to the obligation's obligation type can be used.** It should be noted that only start options related to the obligation's obligation type can be specified. If the obligation is **active**, you may not change its obligation type.

When you click **OK**, a warning window will appear that summarizes the changes that will be made to the obligation. After confirming what will be changed, click **OK** to change the obligation's terms. The **Start Option** that is shown on the **Main** tab will now reflect the start option whose terms were just applied.

## Obligation - Rate Info

The rate info page contains the obligation's rate and information needed by the rate to calculate the obligation's bill segments. This information is only displayed if the obligation's obligation type allows a rate. Generally speaking, rates are utilized by bill based taxes. These include real property taxes in which certain rates are needed for calculations of bill amounts. The local authority that create how and what property is assessed and billed usually supply these rates.

Open **Taxpayer Information, Obligation** and navigate to the **Rate Info** tab to access this page.

### Description of Page

The **Rate** grid controls the rate used to calculate the obligation's bill segments. Multiple rows will only exist if the obligation's **Rate** changes over time.

- An obligation's **Rate** is populated by the system. The obligation type's default rate is populated on the obligation.

Billing selects a single rate from the **Rate** grid when it calculates a bill segment for the obligation. If multiple rates are defined in this grid, the rate is selected based on the obligation's obligation type's Rate Selection Date. The following fields display:

- **Effective Date** is the date the rate becomes effective. The obligation's start date defaults.

- **Rate Schedule** defines the rate used to calculate the obligation's bill segments. Note, you can only choose rates defined as permissible on the obligation's obligation type; the permissible rates are shown in the adjacent tree.

The **Contract Rider** grid contains the contract riders that are in effect for the obligation. This information is defaulted when service is started using a start option (using the contract rider information defined on the start option). You should only have to access this section if you need to fine-tune what the system defaults.

Billing uses this information when it calculates a bill segment for the obligation if:

- The rate used to calculate the bill segment contains a rate component that references a rate factor.
- The referenced rate factor requires a contract rider to levy the associated charge.

If a **Contract Rider** does not exist or is not effective, the charge associated with the rate component is not applied to the taxpayer's bill.

To modify a contract rider linked to the obligation, simply move to a field and change its value. To remove a contract rider, click the - button. To add a new contract rider, click the + button and fill in the information for each field. The following fields display:

- Use **Start Date** to define the date on which the contract rider becomes effective. The obligation's start date defaults.
- Use **End Date** to define the date on which the contract rider expires. This field need only be specified if the contract rider expires on an explicit date.
- Use **Rate Factor** to define the type of rider. You may only reference **Rate Factors** designated as being applicable for contract riders. Note, the **Rate Factors** linked to the rate that are marked with contract rider applicability are shown in the adjacent tree under the Contract Rider node.

For more information about contract riders, refer to [Defining General Rate Factor Information](#).

The **Contract Values** grid contains the contract values that are in effect for the obligation. This information is defaulted when service is started using the contract value information defined on the start option used to start service (if any). You should only have to access this page if you need to fine-tune what the system defaults.

Billing uses this information when it calculates a bill segment for the obligation if:

- The rate used to calculate the bill segment contains a rate component that references a rate factor.
- The referenced rate factor uses a contract value to define the amount charged.

The following fields display:

- Use **Start Date** to define the date on which the contract value becomes effective. The obligation's start date defaults.
- Use **End Date** to define the date on which the contract value expires. This field need only be specified if the value rider expires on an explicit date.

- Use **Rate Factor** to define the type of value. You may only reference **Rate Factors** designated as allowing values in contract terms. Note, the **Rate Factors** linked to the rate that are marked as allowing values in contract terms are shown in the adjacent tree under the Contract Value node.
- Use **Value** to define contract value.

For more information about contract values, refer to [Defining General Rate Factor Information](#).

The **Tax Exemptions** section contains the taxes from which the obligation is all or partially exempt.

Billing uses this information when it calculates a bill segment for the obligation if:

- The rate used to calculate the bill segment contains a rate component that references a rate factor.
- The referenced rate factor can have tax exemptions that limit the amount of the charge.

If the rate factor doesn't have a tax exemption or there are none that are effective during the bill period, the tax is levied in full. If the obligation is exempt, the appropriate charge associated with the tax is calculated. We strongly recommend using the "%T" substitution character in the [Description on Bill](#) on tax-exemptible rate components so that an appropriate tax exemption message appears on the taxpayer's bill.

For more information about tax exemptions, refer to [Defining General Rate Factor Information](#).

The following fields display:

- Use **Start Date** to define the date on which the tax exemption becomes effective.
- Use **End Date** to define the date on which the tax exemption expires.
- Use **Rate Factor** to define the tax from which the obligation is exempt. You may only reference rate factors designated as being tax exemptible. Note, the **Rate Factors** linked to the rate that are marked as tax exemptible are shown in the adjacent tree under the Tax Exemption node.
- Use **Percent Exempt** to define the taxpayer tax exemption percent (e.g., 90 means the taxpayer is 90% exempt and will only pay 10% of the normal tax rate).
- Use **Tax Exempt Type** to specify the type of exemption.
- Use **Tax Certification** if the taxpayer has a tax exemption certificate.

#### Rate Information Tree

The [tree](#) shows the obligation's permissible rates and the related types of contract riders, contract values and tax exemptions that are possible given these rates.

- The Permissible Rates node displays all the rate schedules associated with the obligation's obligation type.
- The Effective Rate Schedule node indicates the rate schedule currently effective.



- The Contract Rider node displays the rate factors linked to the obligation's rate(s) that may have contract riders defined for them. If you expand a node, you can see the individual rate component(s) on which the rate factor is specified. If the obligation has a value defined for the contract rider that is currently in effect, it is also shown.
- The Contract Value node displays all the rate factors linked to the obligation's rate(s) that may have contract values defined for them. If you expand a node, you can see the individual rate component(s) on which the rate factor is specified. If the obligation has a contract value defined that is currently in effect, it is also shown.
- The Tax Exemption node displays all the rate factors linked to the obligation's rate(s) that may have tax exemptions defined for them. If you expand a node, you can see the individual rate component(s) on which the rate factor is specified. If the obligation has a tax exemption defined for the rate factor that is currently in effect, it is also shown.

### Obligation - Chars, Qty & Rec. Charges

This page contains the Characteristics, Contract Quantities and Recurring Charges that are in effect for the obligation.

Use **Taxpayer Information, Obligation** and navigate to the **Chars, Qty & Rec. Charges** tab to open this page.

#### Description of Page

The **Characteristics** grid contains information that describes miscellaneous information about the obligation. This information is defaulted when obligation is created using the characteristics defined on the start option used to start an obligation (if any). You should only have to access this section if you need to fine-tune what the system defaults.

**Note.** You can only choose characteristic types defined as permissible on the obligation record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

The following fields display:

<b>Effective Date</b>	Indicate the date on which the characteristic value becomes effective. The obligation's start date defaults.
<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

The **Contract Quantity** grid contains the contract quantities that are in effect for the obligation. This information is defaulted when service is started using the contract quantities defined on the start option used to start service (if any). You should only have to access this section if you need to fine-tune what the system defaults.

Billing may use this information when it calculates a bill segment for the obligation if the rate has [RQ \(rate quantity\) rules](#).

The following fields appear:

<b>Effective Date</b>	The date this quantity becomes effective. The obligation's start date defaults.
<b>Contract Quantity Type</b>	The type of contract quantity.



**Contract Quantity**

The contract quantity.

The **Recurring Charge** section contains effective-dated information that defines the recurring charge amount used to calculate the obligation's bill segments. This information is only displayed if the obligation's obligation type allows a recurring charge amount.

To modify a recurring charge, simply move to a field and change its value. To change the recurring charge amount effective on a given date, click the + button to insert a row, then fill in the information for each field. The following fields display:

**Effective Date**

The date the charge becomes effective. The obligation's start date defaults.

**Recurring charge changes during a bill period.** Only one recurring charge will be used on any bill segment. The system selects the recurring charge value effective on the end date of the bill segment.

**Recurring Charge Amount**

The recurring charge amount. Recurring charges might include certain administration fees for living or working in a specific tax district (e.g. a special charge for certain school district for each bill sent out)

**Obligation - Miscellaneous**

The miscellaneous page contains information that describes miscellaneous information about the obligation. Use **Taxpayer Information, Obligation, Misc** to open this page.

**Description of Page**

Select the **Industry code** associated with the taxpayer. Some examples of these codes are NAICS and SIC which are used to classify businesses in North America.

Enter a **Business Activity** associated with the taxpayer. This can be used to further define what types of activities are served by this contract.

The bill message grid contains information about the message(s) to appear on an obligation's bill segments. Both one-time and permanent messages may be defined. This information is used when the system assembles messages to appear on an account's bill (as part of bill completion). Refer to [The Source Of Bill Messages](#) for more information. The following fields are displayed:

- For **Bill Message Code**, select a message code to appear on the taxpayer's bill.
- For **Bill Message Type**, use **Temporary** to indicate the message should only be linked to the next bill produced for the account. Use **Permanent** if the message should appear on every bill. Note, a value of **Temporary** defaults.
- The terms and conditions (T&C) grid contains the Obligation's T&C's. This information is defaulted when service is started using a start option (using the T&C information defined on the start option). You should only need to change this information if you need to fine-tune the defaults. Typically T&C is not used frequently with tax authorities. They will probably only be used for certain bill based taxes where very specific items like fees need to be taken into account. The following fields display:
- Use **Start Date** to define the date on which the T&C becomes effective. The Obligation's start date defaults.

- Use **End Date** to define the date on which the T&C expires. This field need only be specified if the T&C expires on an explicit date.
- Use **Terms and Conditions** to define the type of T&C.

## Maintaining Locations

The location record contains geographic information about your service addresses.

The following methods list the various ways in which locations can be created:

- A location can be added via the [location replicator](#).
- A location can be added using this transaction.

The topics in this section describe the location maintenance transaction.

### Contents

[Location - Main Information](#)  
[Location - Characteristics](#)  
[Location - Geographic Data](#)  
[Location - Alternate Address](#)

### Location - Main Information

The Location page contains basic location information. Open **Taxpayer Information, Location** and navigate to the **Main** tab to maintain this information.

#### Description of Page

Basic information about the **Location** and the location's unique identifier (i.e., the **Location ID**) are displayed on every page. These values only appear after the location exists on the database. The ID is a system assigned random number that stays with a location for life.

**Formatting is performed by a plug-in.** The basic information about a location that appears at the top of this page (and on many other pages in the system) is controlled by a plug-in algorithm on the [installation record](#). Refer to the base package's [location format algorithm](#) for an example. If you prefer a different format, your system administrator should configure the system appropriately.

Enter a **Location Type** to categorize the type of location.

The address's constituent fields vary based on the **Country**. Please refer to the [Country](#) page for more information.

**Default note.** A location's **state, city, county, division, characteristics, and geographic data** default from your postal default information. If you change the location's postal code, the system will default geographic values based on the new postal code.

Use **Division** to define the jurisdiction in which the location is located. This defaults based on the **Country** and the **Postal Code**, but can be overridden here.

Indicate whether the address is a valid **Mailing Address**. Locations that are valid mailing addresses may be specified as the mailing location on an account. Refer to [Account - Person Information](#) for more information.

You may reference a **Parent Location** to include this location in a location hierarchy.

At the bottom of this page is a [tree](#) that shows the various objects linked to the location. You can use this tree to both view high-level information about these objects and to transfer to the respective page in which an object is maintained.

## Location - Characteristics

The Characteristics page contains information that controls taxation and other rate options that differ based on geography. Use **Taxpayer Information, Location** and navigate to the **Characteristics** tab to open this page.

### Description of Page

The assessed value of a location and the total obligation amount are controlled by the location's characteristic values (e.g., the taxing city defines the city tax percent applied to the location's value or certain renovations made to the location affect the appraised value). Refer to [An Illustration Of A Rate Factor And Its Characteristics](#) for more information.

**Note.** You can only choose characteristic types defined as permissible on the location record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

The following fields display:

<b>Effective Date</b>	Indicate the effective date of the characteristic type and value. The effective date defaults from the <a href="#">Installation Record</a> when you are adding a new location. The effective date defaults to the current date when you are changing an existing location.
<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

**Default note.** A location's characteristics default from your postal default information. Refer to [Setting Up Location & Service Point Postal Defaults](#) for more information. If you change the location's postal code, the system will default geographic values based on the new postal code.

## Location - Geographic Data

The Geographic Data page contains information that defines where the location is located. Use **Taxpayer Information, Location** and navigate to the **Geographic Data** tab to open this page.

### Description of Page

Enter the **Time Zone** in which the location is located. This value defaults from your [postal defaults](#). The geographic information is used by [Control Central](#) to look for locations. The following fields display:

**Geographic Type**

Indicate the type of geographic data.

**Geographic Value**

Specify the coordinate value. If the entered value must be in a specific format, a description of the required format is displayed adjacent. For example, if you see the format 99A 99A 99 9, you must enter 2 numbers, followed by a letter, followed by a space, followed by 2 numbers, followed by a letter, followed by a space, followed by 2 numbers, followed by a space, followed by a single number.

**Formatting is performed by a plug-in.** The format that is applied to a **Geographic Value** is controlled by the algorithm that is plugged in on the respective [Geographic Type](#). If you prefer a different format, your system administrator should configure this algorithm appropriately. Note, algorithms of this type will NOT convert the input value into the relevant format (i.e., you must enter the information in the exact format dictated by the algorithm).

**Default note.** A location's geographic data defaults from your postal default information. Refer to [Setting Up Location Postal Defaults](#) for more information. If you change the location's postal code, the system will default geographic values based on the new postal code.

**Location - Alternate Address**

This tab only appears if you have enabled alternate addresses on the installation record. Refer to the description of the Alternate Representation field under [Installation - Main](#) for more information.

This page is used when you have an alternate way to define a location's address. This is typically used in countries that use multiple character sets (e.g., the **Main** address is entered in Chinese, the **Alternate Address** is entered in English). When a location has an alternate address, both the main and alternate addresses can be used to search for a location.

Open **Taxpayer Information, Location** and navigate to the **Alternate Address** tab to maintain this information.

**Description of Page**

The remaining fields on the page are used to define the location's alternate address.

The address's constituent fields vary based on the **Country**. **Country** is always protected on this page because a location's alternate address must be located in the same country as its main address. Please refer to the [Country](#) page for more information about address constituents.

**Default note.** An alternate address's **state**, **city** and **county** default from your postal default information. Refer to [Setting Up Location Postal Defaults](#) for more information.

The **Swap Main / Alt Address** button becomes enabled when you've entered an **Alternate Address**. When clicked, the contents of the address on the **Main** tab are swapped with the **Alternate Address**.

## Addition Location Management Tools

---

### Contents

[Using The Location Replicator](#)  
[Location Management](#)  
[Location Management Page](#)

## Using The Location Replicator

You use the location replicator to create many copies of a location.

Perform the following steps to replicate a location:

- Add a [location](#).
- Use the [location replicator](#) to create copies of the location.

The topics in this section describe how to use the location replicator.

**Warning!** The system warns you if you attempt to create. However, this is just a warning as you may want to generate similar locations and then update them with, for example, a unique apartment number on [Location - Main Information](#).

### Contents

[Location Replicator - Main](#)  
[Location Replicator - Location](#)

## Location Replicator - Main

Use **Taxpayer Information, Location Replicator** to open this page.

### Description of Page

Choose the **Location** that serves as the template location.

**Recommendation.** Carefully verify the template location using the other pages in this page before you save the replicated locations. After you save the replicated locations, any corrections could prove time consuming.

The two sections at the top of the page determine the number of replica locations to be created and the address information that will be set up for each location. The left section is used to define how the system will create the part of the address that is *different* for each replica location. The right section defines the parts of the address that will be the *same* for each replica location.

In the left section:

Use the **Replicate** option to control how the position of the replicated number in the first address line:

- Choose **Street Number** if the replicated number should prefix the **Base** value in the first address line of the location. For example, if you have a **Start #** of **1020** and a **Base** value of **Main St** and you choose the **Street Number** option, the first replicated location will be **1020 Main St**.

- Choose **Apartment Number** if the replicated number should suffix the **Base** value in the first address line of the location. For example, if you have a **Start #** of **1020** and a **Base** value of **101 Main St, Apt.** and you choose the **Apartment Number** option, the first replicated location will be **101 Main St, Apt. 1020**.

Indicate the **Number Of Locations** you wish to create.

Use **Base** to define the information that will appear at the end of the first address line on each of the replicated locations. Use **Start #** to define the number assigned to the first copy of the location. Use **Increment** to define the value to increment successive numbers by. The first and last address lines to be created by the replicator are displayed below.

For example, if the first address you want to create is 101 Derby St, and the last address is 199 Derby and you want to do the odd numbered side of the street, you'd enter the following parameters:

Number of Locations would be **50**

Address Suffix would be **Derby St**

Start From would be **101**

Increment by would be **2**

The right section displays the rest of the address information that will be copied onto each replicated location. This information is defaulted from the template location that you selected. The Address 1 field is not shown because you define the information to be entered there in the previous section, as explained above. The remaining fields, however, may be edited if so desired.

After entering the parameters, click the **Replicate** button to generate your new locations. You can use the sections at the bottom of the page to view the locations that will be created when you click save. The left section will show how Address Line 1 will look on the first 10 locations. If you are creating more than 10 locations, the right section will show how Address Line 1 will look on the last 10 locations. If everything looks clean, click save.

## Location Replicator - Location

The Location page is a display only page provided to let you confirm that the location you selected on the add dialog is the one to be used as the template by the replicator. Use **Taxpayer Information**, **Location Replicator**, **Location** to open this page.

### Description of Page

This is a display only page that displays key information about the location being replicated.

## Location Management

The location management functionality facilitates the following:

- Grouping locations together under a single parent location
- Quick & efficient setup and maintenance of apartment complexes and high-rises
- And more...

The topics in this section describe location management functionality.

## Contents

- Define Location Hierarchy
- Manage Groups of Locations

## Define Location Hierarchy

Location management allows you to define location hierarchy. For example, you could also have the notion of

- A building
  - Multiple floors
    - Multiple apartments per floor

To define this hierarchy you must create a [location](#) for each level in the hierarchy and for each "child" location, indicate its appropriate parent location. Once you have set up your location hierarchies, you may view the hierarchies on trees on the location page and on the control central [location tree](#).

In addition, the system provides the following [alerts](#):

- When viewing a parent location, an alert indicates how many child locations exist
- When viewing a child location, an alert indicates that it is linked to a parent location.

## Manage Groups of Locations

The powerful search criteria on the location management page allows you to view different groups of locations based on any combination of parent location address information and general geographic type information. You may also define search criteria such "locations linked to **any** parent location" or "locations **not** linked to any parent location". For example, you could display all locations in a given latitude / longitude that are not linked to any parent location.

Once your search results display the desired list of locations, you could do a mass update to assign or remove the parent location.

## Location Management Page

This page allows you to display a list of locations using a combination of search criteria. In addition, you can perform actions on one or more of the resulting locations, including:

- Assigning a parent location to one or more locations
- Removing the link to a parent location from one or more locations

Open this page using **Taxpayer Information, Location Management**.

### Description of Page

The top half of the page is where you enter the criteria used to search for locations.

**Multiple search criteria may be specified.** You can search for locations using a combination of search criteria.

**At least one positive criterion is required.** You may not define your search criteria to only use filter values that start with "Not Linked...". At least one "positive" filter value is required.

The following table describes each of the different search methods.

Search Method	Description
Parent Location Filter	<p>Use this filter to narrow down your search based on parent location. Enter one of the following values:</p> <p><i>Not Applicable</i></p> <p><i>Linked To This Parent Location</i> - for this option, you must enter a <b>Location ID</b> for the parent location</p> <p><i>Not Linked To This Parent Location</i> - for this option, you must enter a <b>Location ID</b> for the parent location</p> <p><i>Linked To Any Parent Location</i></p> <p><i>Not Linked To Any Parent Location.</i></p> <p>A filter value of <i>Not Applicable</i> defaults.</p>
Location Filter	<p>Use this filter to narrow down your search based on account. Enter one of the following values:</p> <p><i>Located At This Address</i> - for this option, you must enter address constituents</p> <p><i>Located At This Geographic Type / Value</i> - for this option, you must enter a <b>Geo Type</b> and <b>Value</b></p> <p><i>Not Applicable</i></p> <p><i>Show This Specific Location</i> - for this option, you must enter a <b>Location ID</b></p> <p>A filter value of <i>Not Applicable</i> defaults.</p>

The **Select All** / **Clear All** buttons are used to select locations if you plan on issuing any of the mass update actions at the bottom of the page. Refer to the description of the "mass update" actions below for more information.

**50 locations at a time.** Clicking **Select All** selects the first 50 locations in the grid. If more than 50 locations exist, you must select them in batches.

The grid that follows contains the locations that match your search criteria. The following information appears in the grid:

- **Select box.** Use this checkbox to select locations for mass update actions.
- The **Location Information** column shows information about each location. Click the hyperlink to transfer to the [Location](#) page where you can update information about the location in question.
- The **Parent Location Information** column displays information about the location's parent location, if one exists.

This transaction has sophisticated logic that can be used to perform "mass updates" to the locations that appear in the grid using the buttons at the bottom of the page. The buttons are enabled if you select at least one row from the location grid. The following points describe these mass update actions:



The **Assign Parent Location** button is used to assign a parent location to one or more locations. When clicked, the **Assign Parent Location** window opens. Specify the **Parent Location ID** and click **Assign**.

The **Remove Parent Location** button is used to remove the parent location from one or more locations. When clicked, the parent location ID of all selected locations is reset.

## Setting Up Bill Print Groups

Bill print groups allow you to categorize an account's obligations into groups for bill print purposes.

**Bill print groups are optional.** Typically, only accounts with many obligations will use bill print groups because the standard bill print priority is sufficient for accounts with a limited number of obligations. Refer to [Obligation Type - Billing](#) for more information about the standard bill print priority.

Let's use an example to clarify the bill print group concept. Consider a local government's account. This account would have many obligations (some for the police department, others for the court system, others for the department of public works, etc.).

If you don't create a bill print group for this taxpayer, their bill segments will be printed in the order dictated by each segment's obligation's obligation type's bill print priority. If you create a bill print group for this taxpayer, you can define the taxpayer's desired categories (e.g., police, courts, DPW, etc.) and then link each of the account's obligations to the appropriate category.

**Nomenclature.** We refer to the categories under a bill print group as "subgroups". A bill print group can have an unlimited number of subgroups. When you create a subgroup, you define its relative bill print priority and the associated verbiage to print on the bill (assuming you print information about the subgroup on the bill).

We'd like to highlight the following characteristics of bill print groups:

- A bill print group is associated with a specific account. If multiple taxpayers have the exact categorization preferences, you will have to set up multiple bill print groups.
- Over time, a taxpayer could have many bill print groups. This would happen if a taxpayer's bill categorization preferences change over time. We'd like to stress that while an account can have multiple bill print groups, only one will be used by the bill print process (the one effective on the bill date).
- Bill print groups only affect printed bills. Bill print groups do not affect the order in which obligations appear on the bill maintenance transaction.
- If new obligations are added after a bill print group is set up for an account, they will not be linked to a subgroup. If you neglect to link the new obligations to one of the bill print group's subgroups they will be printed under the "default" subgroup (every bill print group must have one default subgroup to cater for this situation).

**Important!** While the system supports the definition of bill print groups and the categorization of an account's obligations into the various subgroups, the base package's bill print extract does NOT take advantage of this information.

The topics in this section describe how to set up a bill print group and how to link an account's obligations to its subgroups.

## Contents

[Bill Print Group - Main](#)

[Bill Print Group - Obligation Sub Group](#)

## Bill Print Group - Main

Open **Taxpayer Information, Bill Print Group** to maintain an account's bill print group and subgroups. After defining this information, transfer to the **Obligation Sub Group** tab to link the account's obligations to the subgroups.

### Description of Page

The **Bill Print Group ID** is displayed on every page. This value only appears after the bill print group exists on the database. The ID is a system-assigned random number that stays with a bill print group for life.

Enter the **Account ID** associated with the bill print group.

Use **Effective Date** to define when the bill print group is effective for the account. This date is important as it allows a taxpayer to change their preferences over time. For example, if the taxpayer wants to change the number of subgroups on a given date, you would simply add a new bill print group effective on this date and then define the new subgroups (and link each obligation to one of the subgroups).

Use **Status** to define if the bill print group is **Active** or **Inactive**. You would only use the **Inactive** value if the bill print group is no longer needed (as there is no delete action on this transaction).

Enter a brief **Description** of the bill print group.

Use **Comments** to describe anything special about the bill print group.

The grid contains the bill print group's subgroups. The following information should be defined for each subgroup:

- **Bill Print Sub Group.** This is the unique identifier of the bill print subgroup.
- **Sub Group Bill Print Priority.** This is the relative print priority of the subgroup in respect of the other subgroups.
- **Use as Default.** Turn on this switch for the default sub group. A bill print group must have one and only one default subgroup. The default group is used by the bill print process if it detects an obligation that is not linked to a subgroup (it links this obligation to the default subgroup).
- **Sub Group Description.** This is a brief description of the subgroup.
- **Description on Bill.** This is the verbiage that will print on the bill (assuming you print something on your bills for the subgroup).

After defining the bill print group's subgroups, navigate to the **Obligation Sub Group** tab to link the account's obligations to the subgroups.

## Bill Print Group - Obligation Sub Group

This page is used to link the account's obligations to one of the bill print group's subgroups.

Open **Taxpayer Information, Bill Print Group** and navigate to the **Obligation Sub Group** page to maintain this information.

### Description of Page

The **Bill Print Group's ID, Account ID** and **Effective Date** are displayed at the top of the page.

The filters control the obligations that appear in the **Obligations in Bill Print Group** grid. The following points describe the various options:

- Use the **Obligation Filter** to define the types of obligations that appear in the grid. The following options are available:
  - **All**. Use this option if you do not wish to restrict obligations based on obligation attributes.
  - **Bill Print Sub Group**. Use this option to restrict obligations to those that are linked to a given **Bill Print Sub Group**.
  - **Obligation type**. Use this option to restrict obligations to those linked to a given **Division** and **Obligation Type**.
- Use **Status Filter** to restrict the obligations based on their status. The following options are available:
  - **All**. This option shows all obligations regardless of status.

Don't forget to click the search button after changing the filters.

The **Obligations in Bill Print Group** contains an entry for every non-cancelled obligation linked to the account that is linked to one of the bill print group's subgroups. The following information appears in the grid:

- Use **Bill Print Sub Group** to define the subgroup associated with the obligation.
- Use **Sequence** when there is more than one obligation in the subgroup. The sequence controls the order in which the obligation's financial information appears on the bill.
- The **Obligation Information** column provides a brief description of the obligation.
- The **Obligation** column contains the unique identifier of the obligation.
- The **Location Information** column contains the characteristic location associated with the bill segment's obligation.

The next set of filters control the obligations that appear in the **Obligations Not in Bill Print Group** grid. The following points describe the various options:

- Use the **Obligation Filter** to define the types of obligations that appear in the grid. The following options are available:
  - **All**. Use this option if you do not wish to restrict obligations based on obligation attributes.
  - **Obligation type**. Use this option to restrict obligations to those linked to a given **Division** and **Obligation Type**.
- Use **Status Filter** to restrict the obligations based on their status. The following options are available:
  - **All**. This option shows all obligations regardless of status.

Don't forget to click the search button after changing the filters.

The **Obligations Not in Bill Print Group** contains an entry for every non-cancelled obligation linked to the account that is NOT linked to one of the bill print group's subgroups. The following information appears in the grid:

- Use **Bill Print Sub Group** to define the subgroup associated with the obligation.
- Use **Sequence** when there is more than one obligation in the subgroup. The sequence controls the order in which the obligation's financial information appears on the bill.
- The **Obligation Information** column provides a brief description of the obligation.
- The **Obligation** column contains the unique identifier of the obligation.
- The **Location Information** column contains the characteristic location associated with the bill segment's obligation.

## The Big Picture Of Customer Contacts

---

Customer contacts are used to record when and why a taxpayer contacted your company. This information is used for both audit and statistical purposes. The topics in this section provide details about how customer contacts are created and used in the system.

### Contents

- [Customer Contacts Are Associated With Persons, NOT Accounts](#)
- [How Customer Contacts Are Created](#)
- [Customer Contacts Are Used To Trigger Letters](#)
- [A Customer Contact May Trigger Reminders](#)
- [Customer Contacts Can Be Used As Case Files](#)
- [Maintaining Customer Contacts](#)

## Customer Contacts Are Associated With Persons, NOT Accounts

Customer contacts are associated with a Person, not with an Account. To associate a customer contact record with an account, you must select a person related to that account. Unless you have specific reasons to do otherwise, the main taxpayer is usually your best choice.

## How Customer Contacts Are Created

You can add a Customer Contact at any time using either of the following methods:

- Use the [customer contact maintenance](#) page.
- Use the [Customer Contact Zone](#) on the dashboard.

In addition to manually adding customer contacts, your implementation team can set up plug-in algorithms that create customer contacts when certain events transpire. For example, you can plug-in an algorithm on the [Case Type](#) object that creates a customer contact whenever a case enters a particular status. Because the number and type of plug-ins can be customized for your implementation, we cannot provide a concise list of all such algorithms.

In addition to the above, a variety of events can trigger the creation of a customer contact. Refer to [Customer Contacts Are Used To Trigger Letters](#) for the details.

## Customer Contacts Are Used To Trigger Letters

In order to send a letter to a taxpayer, a customer contact must be created for the taxpayer. These types of customer contacts reference a customer contact type that, in turn, references a [letter template](#). The letter template controls the type of information that is merged into the “form letter” and how the letter is physically produced. Refer to [Printing Letters](#) for more information about letter templates and how letters are physically produced.

Customer contacts that trigger letters can be produced by any of the method described under [How Customer Contacts Are Created](#). While a user can create this type of customer contact (e.g., if a taxpayer requests a form to sign-up for automatic payment), the primary sources of customer contacts that trigger letters are via system events and algorithms. Your implementation team can introduce plug-in algorithms to create a customer contact when certain events take place. If the created customer contact references a letter template, a letter will be triggered. Because the number and type of plug-ins can be customized for your implementation, we cannot provide a concise list of all such algorithms.

## A Customer Contact May Trigger Reminders

If you want the system to remind you (via a [To Do entry](#)) about a taxpayer-related issue, you can set up a “reminder” on a customer contact. To do this, follow these steps:

- Determine if you need a new customer contact or if you can reuse one that already exists. If you need a new customer contact, use any of the “online methods” described under [How Customer Contacts Are Created](#).
- Add an entry to the [customer contact's log](#). This log entry should be set up as follows:
  - If the To Do entry should be addressed to a specific user, choose a **Reminder** type of **Send to User** and enter the user's **User ID**.
  - If the To Do entry should be addressed to a group of users, choose a **Reminder** type of **Send to Role** and enter the user group's **To Do Role**.
  - Use **Trigger Date** to define the latest date on which the To Do entry should be created. The reason we indicated this should be the latest date is because the background process that's responsible for creating these To Do entries has a parameter called “lead time”. This parameter is used to define the number of days before the **Trigger Date** that the To Do entry should be created. Note, the batch control ID of **TD-CCCB** is used to refer to this background process.

A To Do entry will then be created on a future date.

## Customer Contacts Can Be Used As Case Files

If a taxpayer calls with a complaint (e.g., they suspect their bill is too high), you can start a “case file” for the complaint by creating a customer contact. This customer contact should be marked as “open”.

**Alerts.** Customer contacts that are marked as being open cause an alert to appear when the taxpayer is displayed on control central.

Over time, as you work on the case, you can add entries to the [customer contact's log](#) to describe your progress.

You can use the customer contact's [reminder](#) functionality to remind you to check out the case on a future date (or dates).

When the case is resolved, you simply turn off the customer contact's open indicator and add an appropriate entry to the log describing the resolution.

## Maintaining Customer Contacts

Customer contacts are used to record when and why a customer contacted your company. This information is used for both audit and statistical purposes. The topics in this section describe how to maintain customer contacts. Refer to [The Big Picture Of Customer Contacts](#) for background information about customer contacts.

### Contents

- [Customer Contact - Main](#)
- [Customer Contact - Log](#)
- [Customer Contact - Characteristics](#)

### Customer Contact - Main

Open **Taxpayer Information, Customer Contact** to maintain a customer contact.

**The dashboard.** A zone exists in the [dashboard](#) that can be used to easily create new customer contacts.

### Description of Page

Enter the **Person ID** of the person associated with the customer contact.

Turn on the **Open** switch if the event that necessitated the contact hasn't been resolved. For example, if a taxpayer calls with a high bill complaint and you can't resolve it immediately, you would turn on the **Open** switch and enter an appropriate entry in the **Log**. Refer to [Customer Contacts Can Be Used As Case Files](#) for more information.

Enter the **Contact Date** and **Contact Time**.

**Default note.** The current date and time are defaulted.

The **User ID** of the user who created the contact is displayed adjacent to **Contact Date / Time**.

Every customer contact has a **Contact Type** that classifies the record for reporting purposes. Every contact type, in turn, references a **Contact Class**. The class categorizes customer contacts into larger groupings for reporting purposes.

**Adding a customer contact may cause a letter to be generated.** You can set up a customer contact type to generate a form letter whenever a customer contact of this type is added. In fact, this is the only way to generate a letter in the system. Refer to [Customer Contacts Are Used To Trigger Letters](#) for more information.

Use **Comments** to describe the contact.

If a letter template is associated with the customer contact type / class:

- **Letter Information** describes the status of the letter (i.e., whether it has been printed or not).
- If the letter has been printed and you want to reprint, please turn on the reprint switch (this will cause the letter to be reprinted the next time the respective letter print background process executes).
- If your implementation team has plugged-in an Online Letter Image algorithm on [Installation Options - Algorithms](#), the **Display Letter** button appears. When clicked, the image of the letter is rendered in a PDF and displayed in an Adobe reader. Refer to [Technical Implementation Of Online Letter Production](#) for a technical description of how letter images are produced.

The grid that follows contains a diary of past (and future) events related to the customer contact. Refer to [A Customer Contact May Trigger Reminders](#) and [Customer Contacts Can Be Used As Case Files](#) for more information about when you would use this log.

There are two ways to add a row to the log:

- You can click the + button to add a new row.
- You can navigate to the **Log** tab and insert a new row into the scroll.

Regardless of the method used to add a log entry, the following information appears on a log entry:

- **Create Date / Time** is the date and time when the log entry was created.
- **Created by** identifies the user that created the log entry.
- Use **Comments** to describe the reason for the log entry. If you have lengthy comments, we recommend navigating to the **Log** tab (by clicking the adjacent Go To button) as there is a larger input field on this page.
- The remaining fields are only used if you want the system to remind you about this customer contact on a future date. If you enter these fields, the system will create a To Do entry to remind you about the customer contact.
  - If the To Do entry should be addressed to a specific user, choose a **Reminder** type of **Send to User** and enter the user's **User ID**.
  - If the To Do entry should be addressed to a group of users, choose a **Reminder** type of **Send to Role** and enter the user group's **To Do Role**.
  - Use **Trigger Date** to define the latest date on which the To Do entry should be created. The reason we indicated this should be the latest date is because the background process that's responsible for creating these To Do entries has a parameter called "lead time". This parameter is used to define the number of days before the **Trigger Date** that the To Do entry should be created. Note, the batch control ID of **TD-CCCB** is used to refer to this background process.

**Multiple reminders.** You can set up multiple reminders on a customer contact. For example, you can indicate that you want to be reminded every Monday for the next 4 weeks to check on the issue that caused a given customer contact to arise. You'd do this by entering 4 log entries where each has the desired **Trigger Date**.



## Customer Contact - Log

The log contains a diary of past (and future) events related to the customer contact. Open **Taxpayer Information, Customer Contact** and navigate to the **Log** tab to maintain a customer contact's log. Note, you can also maintain this information on the **Main** tab.

Refer to [A Customer Contact May Trigger Reminders](#) and [Customer Contacts Can Be Used As Case Files](#) for more information about when you would use this log.

### Description of Page

**Date / Time** is a display-only field that contains the date and time when the log entry was created.

**Log ID** is the system-assigned, unique identifier of the log entry.

**Created by** is a display-only field that contains the ID and name of the user who created the log entry.

Use **Comments** to describe the reason for the log entry.

The remaining fields are only used if you want the system to remind you about this customer contact on a future date. If you enter these fields, the system will create a To Do entry to remind you about the customer contact.

- If the To Do entry should be addressed to a specific user, choose a **Reminder** type of **Send to User** and enter the user's ID in **Send To**.
- If the To Do entry should be addressed to a group of users, choose a **Reminder** type of **Send to Role** and enter the user group's To Do Role in **Send To**.
- Use **Trigger Date** to define the latest date on which the To Do entry should be created. The reason we indicated this should be the latest date is because the background process that's responsible for creating these To Do entries has a parameter called "lead time". This parameter is used to define the number of days before the **Trigger Date** that the To Do entry should be created. Note, the batch control ID of **TD-CCCB** is used to refer to this background process.

## Customer Contact - Characteristics

Open **Taxpayer Information, Customer Contact** and navigate to the **Characteristic** tab to maintain a customer contact's characteristics.

**Note.** We do not anticipate a great deal of use for characteristics on customer contacts. The primary purpose of catering for characteristics on this object is to allow a [customer contact that triggers a letter](#) to be able to extract information from any object in the system. For example, if you have an algorithm / background process that wants to generate a letter in respect of a given location, you could create a customer contact and reference the location as a characteristic value. Then, when the information is extracted for the letter, the extract algorithm could extract fields from the location to merge onto the letter.

### Description of Page



**Note.** You can only choose characteristic types defined as permissible on a customer contact. Refer to [Setting Up Characteristic Types & Their Values](#) for more information about characteristic types. In addition, the characteristic type must also be defined as valid for the customer contact's type. Refer to [Setting Up Customer Contact Types](#) for more information about customer contact type.

The following fields display:

<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

## Printing Letters

As described under [Customer Contacts Are Used To Trigger Letters](#), you must create a customer contact whenever you want to send a letter to a taxpayer. The contents of this section describe the technical implementation of letter production.

### Contents

- [Letter Templates Control The Information Merged Onto Letters](#)
- [Technical Implementation Of Online Letter Images](#)
- [Technical Implementation Of Batch Letter Production](#)
- [Reproducing The Letter Print Flat File](#)
- [How To Reprint A Specific Letter](#)

## Letter Templates Control The Information Merged Onto Letters

Customer contacts that trigger letters reference a [customer contact type](#) that, in turn, references a [letter template](#). The letter template controls the following:

- It contains an algorithm that is responsible for extracting the information merged onto your letters. Specifically, algorithms of this type create the "flat file records" that are passed to your letter print software. Algorithms of this type are called under the following scenarios:
  - The [background process](#) that builds the flat file that's passed to your letter print software calls these algorithms to construct the "flat file records" for each letter.
  - If your letter print software has the ability to construct a real-time image of a letter (in a PDF), you can plug-in an **Online Letter Image** algorithm on the [Installation Record](#). This type of algorithm will call the letter's letter template's extract algorithm to extract the information that is merged onto the letter. Refer to [Technical Implementation Of Online Letter Image](#) for the details.
- It contains the ID of the background process that builds the flat file that's passed to your letter print software. The base package example of this process (known by the batch control ID of [LTRPRT](#)) simply calls each customer contact's letter template's extract algorithm to format the information placed onto the flat file.

## Technical Implementation Of Online Letter Images

Users can view an image of any letter that is sent to a taxpayer if you set up the following:

- Plug-in an **Online Letter Image** construction algorithm on the [Installation Record](#). Refer to [ONLD-LT](#) for an example of such an algorithm. Note, if your letter print software is not capable of producing a PDF containing an image of a letter, users will not be able to view images of letters.
- Plug-in the appropriate extract algorithm on each [letter template](#). Algorithms of this type format the records that contain the taxpayer information that is passed to your printing software.

When you plug-in these algorithms, a button appears on [Customer Contact - Main](#) for customer contacts whose customer contact type references a letter template. When a user clicks this button, the following takes place:

- The installation record's **Online Letter Image** construction algorithm is executed.
- This algorithm calls the customer contact's letter template's extract algorithm. This algorithm constructs the information that's merged onto the letter and returns it to the **Online Letter Image** algorithm. This algorithm, in turn, passes it to your letter print software.
- Your letter print software renders an image of the letter in a PDF and returns it to the **Online Letter Image** algorithm.
- And finally, the **Online Letter Image** algorithm displays the PDF in a separate Adobe session.

## Technical Implementation Of Batch Letter Production

The batch process that extracts letter information (known by the batch control ID of [LTRPRT](#)) reads all customer contact records in a given run number that are marked with its batch control ID. For each customer contact, it creates numerous records on a flat file. These records contain the taxpayer information that is merged onto your letters.

The information that is placed on each record is controlled by the customer contact's letter template's extraction algorithm. Refer to [Letter Templates Control The Information On Letters](#) for more information.

The records on the flat file can be constructed in either a fixed-position or field-delimited format. The format that's used is controlled by a parameter supplied to the **LTRPRT** background process.

**Print / routing methods.** If you require different print formats, you must create new letter extract algorithms and plug them in on your [letter template](#).

## Reproducing The Letter Print Flat File

You can reproduce the flat file containing the information sent to your printing software at any time. Simply request the respective extract batch process and specify the run number associated with the historic run.

## How To Reprint A Specific Letter

If you need to reprint a specific letter, navigate to [Customer Contact - Main](#) and turn on the **Reprint** switch (and save the customer contact). Alternatively, if your implementation has [enabled the online creation of letter images](#), you can also click the **Display Letter** button on this page and then print the resultant PDF on your local printer.

# Processing Forms

## Contents

- [Background Topics](#)
- [Maintaining Registration Forms](#)
- [Maintaining Tax Forms](#)
- [Maintaining Form Batch Headers](#)

## Background Topics

---

The topics in this section provide background information about form processing functionality.

## Contents

- [Form Type](#)
- [Creating Tax Forms](#)
- [Tax Form Log Information](#)
- [Creating Registration Forms](#)
- [Registration Form Log Information](#)
- [Creating Form Batch Headers](#)
- [Form Batch Header Log Information](#)

## Form Type

Form Type controls the tax form or registration form that references it. Some aspects of the tax form / registration form that form type defines include:

- The effective period of the form type
- The parameters used in forms processing logic.
- The related registration form / tax form business object that defines the structure, lifecycle, processing rules and processing options.

**Your implementation defines your forms and the rules for processing these forms.** Refer to [Defining Form Processing Options](#) for details.

## Creating Tax Forms

To create a tax form, navigate to the Tax Form Portal in add mode, select the form type, fill in the form details and click the save button. Refer to [Maintaining Tax Forms](#) for a detailed description of the Tax Form Portal.

Tax forms can also be created as part of form batch headers. Refer to [Creating Form Batch Headers](#) for a detailed description of the Form Batch Header Portal.

## Tax Form Log Information

A tax form log contains an entry for every recorded event during the lifecycle of a tax form. There are two general types of log entries:

- **Automatic entries.** The system automatically creates an entry in the log when a tax form is created or there is a status change or when a related entity is created. This also includes any implementation-specific log entries. Users cannot modify or delete these log entries.
- **Manual entries.** Users can add manual entries to record significant events at their discretion.

## Creating Registration Forms

To create a registration form, navigate to the Registration Form Portal in add mode, select the form type, fill in the form details and click the save button. Refer to [Maintaining Registration Forms](#) for a detailed description of the Registration Form Portal.

Registration forms can also be created as part of form batch headers. Refer to [Creating Form Batch Headers](#) for a detailed description of the Form Batch Header Portal.

## Registration Form Log Information

A registration form log contains an entry for every recorded event during the lifecycle of a registration form. There are two general types of log entries:

- **Automatic entries.** The system automatically creates an entry in the log when a registration form is created or there is a status change or when a related entity is created. This also includes any implementation-specific log entries. Users cannot modify or delete these log entries.
- **Manual entries.** Users can add manual entries to record significant events at their discretion.

## Creating Form Batch Headers

To create a form batch header, navigate to the Form Batch Header Portal in add mode, select the form batch header type and click the save button. Refer to [Maintaining Form Batch Headers](#) for a detailed description of the Form Batch Header Portal.

## Form Batch Header Log Information

A form batch header log contains an entry for every recorded event during the lifecycle of a form batch header. There are two general types of log entries:

- **Automatic entries.** The system automatically creates an entry in the log when a form batch header is created or there is a status change or when a related entity is created. This also includes any implementation-specific log entries. Users cannot modify or delete these log entries.
- **Manual entries.** Users can add manual entries to record significant events at their discretion.

## Maintaining Registration Forms

---

Use the Registration Form transaction to view and maintain registration forms. Navigate using **Main Menu, Forms, Registration Form**.

### Contents

- [Registration Form Query](#)
- [Registration Form Portal](#)

## Registration Form Query

Use the [query portal](#) to search for an existing registration form. Once a registration form is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Registration Form Portal

This portal appears when a registration form has been selected from the Registration Form Query portal.

The topics in this section describe the base-package zones that appear on this portal.

### Contents

- [Registration Form Actions](#)
- [Registration Form](#)
- [Registration Form Log](#)

## Registration Form Actions

This is a standard [actions zone](#).

If the registration form is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

## Registration Form

The Registration Form zone contains display-only information about the registration form.

Please see the zone's help text for information about this zone's fields.

If the registration form is in a 'suspended' or 'waiting' status, this zone can display the list of issues encountered during form validation. The list includes a 'go to' button that allows you to go directly to the field in error. Re-post the form after resolving all issues.

## Registration Form Log

This is a standard [log zone](#).

## Maintaining Tax Forms

---

Use the Tax Form transaction to view and maintain tax forms. Navigate using **Main Menu, Forms, Tax Form**.

**Contents**

[Tax Form Query](#)  
[Tax Form Portal](#)

## Tax Form Query

Use the [query portal](#) to search for an existing tax form. Once a tax form is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Tax Form Portal

This portal appears when a tax form has been selected from the Tax Form Query portal.

The topics in this section describe the base-package zones that appear on this portal.

**Contents**

[Tax Form Actions](#)  
[Tax Form](#)  
[Tax Form Log](#)

### Tax Form Actions

This is a standard [actions zone](#).

If the tax form is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

### Tax Form

The Tax Form zone contains display-only information about the tax form.

Please see the zone's help text for information about this zone's fields.

If the tax form is in a 'suspended' or 'waiting' status, this zone can display the list of issues encountered during form validation. The list includes a 'go to' button that allows you to go directly to the field in error. Re-post the form after resolving all issues.

### Tax Form Log

This is a standard [log zone](#).

## Maintaining Form Batch Headers

---

Use the Form Batch Header transaction to view and maintain form batch headers. Navigate using **Main Menu, Forms, Form Batch Header**.

**Contents**

[Form Batch Header Query](#)  
[Form Batch Header Portal](#)

## Form Batch Header Query

Use the [query portal](#) to search for an existing form batch header. Once a form batch header is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Form Batch Header Portal

This portal appears when a form batch header has been selected from the Form Batch Header Query portal.

The topics in this section describe the base-package zones that appear on this portal.

### Contents

- [Form Batch Header Actions](#)
- [Form Batch Header](#)
- [Included Forms](#)
- [Form Batch Header Log](#)

### Form Batch Header Actions

This is a standard [actions zone](#).

If the form batch header is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

### Form Batch Header

The Form Batch Header zone contains display-only information about the form batch header.

Please see the zone's help text for information about this zone's fields.

### Included Forms

This zone lists the forms that are a part of the selected form batch header.

### Form Batch Header Log

This is a standard [log zone](#).



# Penalty and Interest

Penalty and Interest or "P&I" is a generic term used to describe a wide range of penalty, interest, and fee liabilities that are calculated and imposed by a tax authority.

## Contents

[Background Topics](#)  
[Maintaining Waivers](#)

## Background Topics

---

The topics in this section provide background information about penalty and interest functionality.

## Contents

[Dynamic Credit Allocation](#)  
[Bringing Penalty and Interest Up to Date](#)  
[Waiving Penalty and Interest](#)

## Dynamic Credit Allocation

Credit allocation is the process of taking credits within an obligation and applying business rules to allocate them against tax, penalty, interest, and fees (in other words, allocated by debt category).

We use the term 'dynamic credit allocation' to indicate that credit allocation should be re-evaluated any time penalty and interest is calculated, whether for updates or forecasting.

The base product provides zones on Control Central on both the [Account Information](#) portal and the [Taxpayer Information](#) portal that show the balances by debt category at various levels (person, account, obligation). Refer to the details of the zone for more information.

**Configuration required.** Dynamic credit allocation must be configured for an obligation. Refer to The [Big Picture of Credit Allocation](#) for more information.

## Bringing Penalty and Interest Up to Date

Depending on how your system is configured, several business processes may force the recalculation of penalty and interest real-time. Some examples of events in the system that may have been configured to bring P&I up to date are as follows:

- When a [tax form](#) posts or is reversed, transferred or adjusted. Basically any state transition of a form that causes adjustments to be created should include a step to bring P&I up to date.
- When a [payment](#) is frozen or canceled
- When the [effective date](#) of a frozen payment is changed
- An [obligation](#)'s override due date changes.
- When a [waiver](#) is activated or canceled

- An event in an [overdue process](#)

In addition to real-time requests to bring P&I up to date, the system provides a background process that may periodically recalculate penalty and interest for all obligations governed by P&I.

**Calculation through date.** Every time the system recalculates P&I for an obligation its calculation through date is updated. This allows a user to know the accuracy of the balance viewing the details provided by [dynamic credit allocation](#).

## Waiving Penalty and Interest

The tax office generally has broad discretionary power to waive penalty, interest and fees. Waivers may be for part of the amount imposed or the full amount and may either be one-time or ongoing.

One-time waivers are typically used for hardship and ongoing waivers are generally used where there has been a natural disaster, or there is an ongoing error or issue with the account that is being researched.

Refer to [Waivers](#) for more information about configuring waivers.

### Contents

- [Waiver Type](#)
- [Creating and Activating a Waiver](#)
- [Canceling a Waiver](#)
- [Waiver Log Information](#)

## Waiver Type

Each waiver has an associated [waiver type](#). The waiver type defines the configuration information that is common to waivers of a given type.

## Creating and Activating a Waiver

A waiver is typically added in **pending** status allowing a user to configure the attributes of the waiver such as the assessment, the debt category and the amount (for a one-time waiver). The waiver has no financial effect at this time.

When the user is happy with the waiver's setup and activates it, the obligation's penalty and interest should be recalculated accordingly.

For any penalty and interest charge that is waived, an adjustment is created for the waived amount (to credit the charge being waived). This information is visible on the [waiver](#) portal.

**Controlled by configuration.** The actual behavior of the waiver and its lifecycle are controlled by the configuration of the waiver. The above information describes the logic provided for waivers in the base product. Your implementation may introduce a different business process.

## Canceling a Waiver

If a waiver was created in error or created incorrectly, it may be canceled. At this point the obligation's penalty and interest should be recalculated again to cancel the waiver financial transactions.

**Controlled by configuration.** The actual behavior of the waiver and its lifecycle are controlled by the configuration of the waiver. The above information describes the logic provided for waivers in the base product. Your implementation may introduce a different business process.

## Waiver Log Information

A waiver log contains an entry for every recorded event during the lifecycle of a waiver. There are two general types of log entries:

- **Automatic entries.** The system automatically creates an entry in the log when a waiver is created or there is a status change or when a related entity is created. This also includes any implementation-specific log entries. Users cannot modify or delete these log entries.
- **Manual entries.** Users can add manual entries to record significant events at their discretion.

## Maintaining Waivers

---

Use the Waiver transaction to view and maintain pending or historic waivers. Navigate using **Main Menu, Financial, Waiver**.

### Contents

[Waiver Query](#)  
[Waiver Portal](#)

## Waiver Query

Use the [query portal](#) to search for a waiver. Once a request is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Waiver Portal

This portal appears when a waiver has been selected from the Waiver Query portal.

The topics in this section describe the base-package zones that appear on this portal.

### Contents

[Waiver Actions](#)  
[Waiver](#)  
[Existing Charges](#)  
[Existing Waivers](#)  
[Waiver Log](#)

## Waiver Actions

This is a standard [actions zone](#).

If the activity request is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

### **Waiver**

The Waiver zone contains display-only information about the selected waiver.

Please see the zone's help text for information about this zone's fields.

### **Existing Charges**

This info zone displays all existing adjustments that have the same Assessment Id and Debt Category as the waiver being displayed.

### **Existing Waivers**

This info zone displays all existing waiver adjustments created for this waiver.

### **Waiver Log**

This is a standard [log zone](#).

# Billing

In this section, we describe how to manage your taxpayer's bills.

## Contents

- [The Big Picture of Billing](#)
- [Maintaining Bills](#)
- [Financial Transactions On A Bill](#)
- [Maintaining Bill Segments](#)
- [Multi Cancel/Rebill](#)
- [Obligation Billing History](#)
- [Bill Exception](#)
- [Bill Segment Exception](#)
- [Maintaining Billable Charges](#)
- [Uploading Billable Charges](#)

## The Big Picture of Billing

Billing is one method used to establish tax assessments for bill-based tax types. Examples of these tax types include real property and vehicle taxes. Many obligation types will not use billing functionality.

Refer to [Defining Obligation Types](#) for a description of the different obligation types.

The billing process is used when the tax authority has the information required to calculate and assess tax for a taxpayer, in contrast to tax types where the taxpayer must file a tax return.

Overdue processing is used to govern collections processing, including the sending of collections letters to taxpayers with past due obligations.

Refer to [Overdue Financial Obligations](#) for more information on overdue processing.

The topics in this section provide background information about a variety of billing topics.

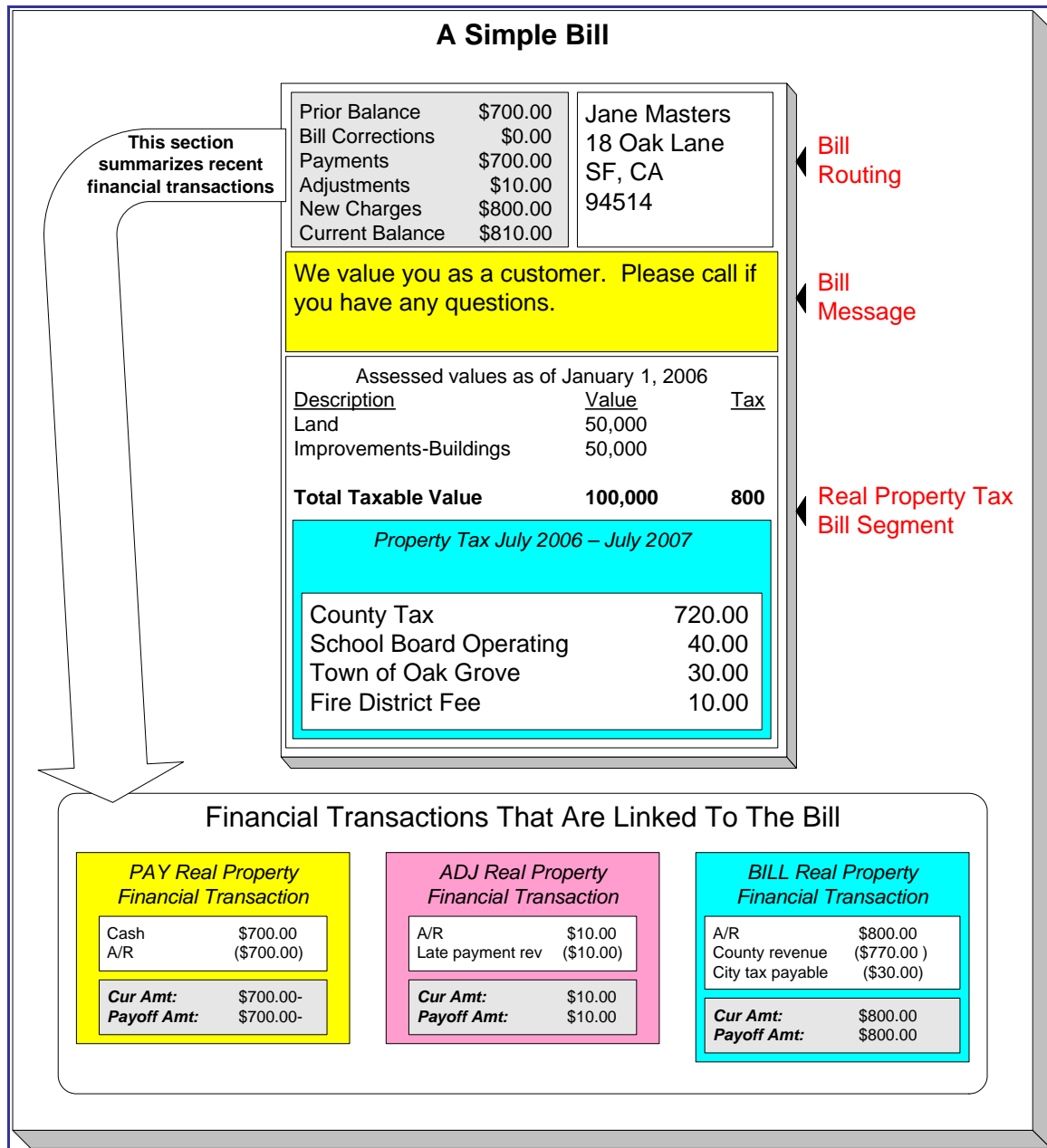
## Contents

- [An Illustration Of A Simple Bill](#)
- [A High Level Overview Of The Bill Creation Process](#)
- [Bill Errors](#)
- [Cancel / Rebill Incorrect Bill Segments](#)
- [How Rates Affect The Information On Bill Segments](#)
- [Bill Frequency - Bill Cycle vs Bill Segment Duration](#)
- [Prorating Charges When a Rate is Applied](#)
- [Batch Billing](#)
- [Billing Financial Transaction Considerations](#)
- [The Source Of Bill Routing Information](#)
- [Bill Messages](#)
- [A Bill May Affect More Than Just Taxpayer Balances](#)
- [Using Billable Charges for Pass Through / Convergent Billing](#)
- [Printing Bills](#)

## Idiosyncratic Manual Bill Cancellation

## An Illustration Of A Simple Bill

The following diagram illustrates a simple bill:



The following concepts are illustrated above:

**A bill is produced for an account**

A bill is produced for an account. Over time, an account receives many bills.

- A bill summarizes financial transactions** A bill contains information about the various financial transactions that have taken place since the last bill was produced (i.e., payments, adjustments, and bill corrections). The above illustration shows a bill with financial transactions for new charges, a payment, and an adjustment.
- A bill is routed to persons** A copy of the bill is sent to every person linked to the account who requires a copy of the bill.
- A bill contains messages** A bill may contain messages.
- A bill typically contains bill segments** A bill typically contains one bill segment for every active billable obligation linked to its account.
- A bill segment contains calculation details** A bill segment contains information showing how the segment was calculated and how it should be printed on the taxpayer's bill.

## A High Level Overview Of The Bill Creation Process

When the system is asked to produce a bill for an account, it attempts to create one or more bill segments for every non-cancelled / non-closed billable obligation linked to the account. Whether or not an obligation contributes bill segment(s) to a bill is a complicated subject as the system supports a wide variety of bill segment creation methods.

The system determines the bill segment creation method to use from the bill segment type on the obligation's obligation type.

Some bill segment creation methods may apply the obligation's rate to a specific number / amount in order to calculate how much the taxpayer owes. The details of the calculations are captured in the bill calculation lines. For more information, refer to [How Rates Affect The Information On Bill Segments](#) for the details. Refer to [Effective Dates & Proration](#) for information about how the system prorates changes to rates and prices during a bill period.

If errors are detected during the bill segment creation process, the bill segment is saved with its error. The system then proceeds to the account's next obligation. This way, a user can see all problematic bill segments so they can be corrected en masse. Refer to [Bill Errors](#) for information about how to deal with bill errors.

When every segment on a bill is error free, the bill is ready to be completed. When a bill is completed, the system:

- Creates a bill routing for each person linked to the account who requires a copy of the bill. The routing information controls the format of the printed bill and how the bill is sent to the person. Refer to [The Source Of Bill Routing Information](#) for the details.
- Links all bill messages to the bill. A bill message can come from a variety of sources. Refer to [The Source Of Bill Messages](#) for the details.
- If the **Freeze At Bill Completion** [installation option](#) has been turned on:
  - All **freezable** bill segments will be frozen.
  - All **freezable** adjustments whose adjustment type indicates **Freeze At Bill Completion** will be frozen.

- Sweeps recent financial transactions that have been created since the last bill was completed.
- There are several other functions that happen at completion time. Refer to the description of the Complete button under [Bill Lifecycle](#) for the details.

And that's it; the bill can now be sent to the taxpayer.

The remaining topics in this section provide more information about the creation and completion of bills.

**Batch and real-time bill creation.** Anything the batch bill process does for whole sets of accounts, you can do to a specific account on-line / real time. Refer to [How To Create A Bill For All Obligations Linked To An Account](#) for information about how to create a bill on-line / real time. Refer to [Batch Billing](#) for more information about the batch bill creation process.

## Bill Errors

The topics in this section describe errors that are detected when the system creates a bill.

### Contents

- [Bill Segment Errors](#)
- [Bill Completion Errors](#)

### Bill Segment Errors

If you consider the huge number of variables involved in the creation of a bill segment, it probably won't surprise you to learn that occasionally, some bill segments are in error.

Errors tend to be caused by missing or inconsistent data. Some examples of classic errors:

- **Missing master data.** For example, if a bill is supposed to be routed to the account's mailing address and the account doesn't have such an address, an error is generated.
- **Missing rate data.** For example, if a rate contains a taxpayer specific charge and there is no value defined on the taxpayer's obligation, an error is generated.

The system saves bill segments that are in error just as it saves bill segments that are error-free. This is done because bill segments are nothing more than a snapshot of the data that was used to calculate the charges. By saving the snapshot, you can see the information the system used when it detected the error and therefore more effectively fix the cause of the error. The standard way to fix an error is to:

- Look at the bill segment to determine the cause of the error.
- Correct the cause of the error.
- Regenerate the bill segment. Regeneration simply deletes the offending segment and recreates a new segment using the corrected information.

For every bill segment in error, a record is written to the [Bill Segment Exception](#) table.

Refer to [How To Correct A Bill Segment That's In Error](#) for instructions describing how to correct a bill segment.



It's obvious but worth stressing that a bill may contain some segments that are perfect and others that are in error. Once all segments that are in error are corrected, the bill can be completed (and sent to the taxpayer).

If the bill segment in error was created as part of the batch bill process, the system attempts to fix the offending segment(s) when cyclical billing runs again by regenerating it. Therefore, if the cause of the error is fixed during the day, the system will automatically regenerate the bill segment when batch billing next runs; you don't have to manually correct each bill. And once a bill is error-free, it will be completed and sent to the taxpayer.

Refer to [Batch Billing](#) for more information.

## Bill Completion Errors

In addition to errors on bill segments, there may also be errors detected when the system attempts to complete a bill. For example, if the system cannot find a mailing address, the bill will be in error (as opposed to one of its bill segments).

Errors tend to be caused by missing or inconsistent data. Some examples of classic errors:

- **Missing mailing address.** For example, if the system cannot find an address to which the bill can be routed, it generates an error.
- **Bill segments are in error.** If a bill contains bill segments that are in error, it cannot be completed and an error is generated.

The system saves bills that are in error just as it saves bills that are error-free. This is done because bills are nothing more than a snapshot of the data that was used to calculate the charges. By saving the snapshot, you can see the information the system used when it detected the error and therefore more effectively fix the cause of the error. The standard way to fix an error is to:

- Look at the bill to determine the cause of the error.
- Correct the cause of the error.
- Recomplete the bill.

For every bill in error, a record is written to the [Bill Exception](#) table.

Refer to [How To Correct A Bill That's In Error](#) for more information.

If the bill completion error was created as part of the batch bill process, the system attempts to recomplete the bill at the next cyclical billing run. Therefore, if the cause of the error is fixed during the day, the system will automatically complete the bill when batch billing next runs; you don't have to manually complete each bill.

Refer to [Batch Billing](#) for more information.

## Cancel / Rebill Incorrect Bill Segments

Sometimes the error on a bill segment is not detected by the system. Such errors occur when the data used to calculate the bill is valid, but wrong. For example, the sales tax percent was entered incorrectly on a rate factor.

There is no way the system can detect such problems and therefore the system freezes the bill segment and routes the bill to the taxpayer. To correct such a bill segment, you must cancel the offending segment and create a new segment (after correcting the cause of the problem). We refer to this process as cancel / rebill.

Refer to [How To Cancel / Rebill A Bill Segment](#) for more information about how to cancel / rebill a frozen bill segment.

You must use the cancel / rebill process to correct problems on frozen bill segments. This is because a frozen bill segment can be thought of as having been posted to your general ledger (even if the GL interface hasn't run). And once a financial transaction is posted to the general ledger, it can only be removed via a reversal. Contrast this to bill segments that are in error – they can be deleted because they were never posted to the general ledger.

Refer to [Bill Segment Lifecycle](#) for more information about the differences between bill segments that are frozen and those that are in error.

**Bill segments are canceled / rebilled.** It's important to stress that bill segments are canceled and rebilled (as opposed to bills). If every bill segment on a bill is incorrect, you must cancel / rebill each individual bill segment.

**Note.** Using the cancel or cancel/rebill functionality will cause the cancel and rebill details to be swept on to the taxpayer's next bill.

## How Rates Affect The Information On Bill Segments

After a billable quantity has been compiled, the system applies the obligation's rate to the quantity to determine how much to charge the taxpayer. We refer to this process as "rate application".

Refer to [Rates](#) for detailed information about rates.

**Not all obligations use a rate!** The obligation's obligation type defines whether or not the obligation needs a rate. The bill calculation algorithm plugged-in on the bill segment type that's referenced on the obligation's obligation type controls whether or not the system calls the obligation's rate to calculate charges. Refer to [Defining Bill Segment Types](#) for a description of the various bill segment creation algorithms that are supported in the system.

**Your proration choices impact what appears on a bill.** A rate contains a variety of effective-dated information (e.g., the prices are effective-dated, the structure of the rate is effective-dated, etc.). If this effective-dated information changes during a bill period, the system may need to prorate the charges. For example, if the sales tax percentage changes mid-period, the system can prorate the tax change (e.g., 20 days at 6% and 11 days at 6.25%). When you setup a rate, you define exactly how the system handles changes that occur during a bill segment. For example, you can tell the system that sales tax changes should not be prorated. Rather, it should use the value effective at the start / end of the bill period (note, you have several other options). We mention this because your choices have a large impact on how a rate affects the information on bill segments. Refer to [Effective Dates & Proration](#) for more information.

Rate application is a very sophisticated process as it can affect every aspect of a bill segment. The following points describe how rate application works at a high-level:

- The calculation details that were amassed earlier are passed into rate application.
- Next, the [rate schedule's rate quantity rules](#) (if any) are executed. Rate quantity rules are used to calculate rate quantities (RQ) referenced in the rate that cannot be derived from the calculation details. For example, if you have a rate that has a charge based on the number of days in the bill period, a rate quantity rule would be necessary to calculate the number of days in the bill period.
- The system determines which of the rate's [rate versions](#) should be processed. Note, a flag on the [rate schedule](#) allows you to control what happens if multiple rate versions are detected during the bill period. A bill segment "calculation detail" is created for each such rate version. Each calculation detail contains the results of executing the rate version's [rate components](#) (where a separate [bill line](#) is created for each rate component). Each bill line contains:
  - An audit of what was priced.
  - How the printed bill line should look (if the bill lines is printed).
  - How the amount of the bill line should be booked in the general ledger.

**Note.** There are many plug-in spots available on a rate component. These plug-ins can manipulate virtually every aspect of the bill segment. This means if you require functionality that isn't support by the base package rate components, you can build additional rate component plug-ins to do whatever you need.

- After every rate version is processed, rate application returns the calculation details to the process that called it. This point is important as it means that rate application executes in memory (i.e., it does NOT insert the calculation details on the database, rather, it returns them to whatever process called rate application). Because rate application executes in memory, it can be used to perform billing calculations in many parts of the system. For example:
  - Billing uses rate application to calculate the charges that are eventually saved on the bill's bill segments.
  - The [Rate Check](#) transaction calls rate application when you want to check a rate real-time. When you use this transaction, you enter the quantities that are passed into rate application and then rate application shows you the calculation details that result.

Refer to [Rates](#) for more information about how rates are constructed.

## Bill Frequency - Bill Cycle vs Bill Segment Duration

An account's bill cycle defines when the system attempts to create bill segments for the account's billable obligations. The word "attempt" is stressed because an obligation may not have a bill segment on every bill created for its account.

Some examples will help make the point:

- An account may be on a monthly bill cycle (meaning we attempt to create a bill every month for the account) but contain only biannual obligations. However, this is not sensible. Why? Because the system would attempt to create a bill every month for the account's obligations, but only twice during the year would it succeed (because the obligations have a biannual duration).
- An account may be on a biweekly bill cycle (meaning we attempt to create a bill every 2 weeks for the account) and contain a mixture of biweekly, monthly, and quarterly obligations. In this scenario, every two weeks the system would create a bill that contains at least one bill segment (for the biweekly obligation). However, 12 times a year, the bill will contain an additional bill segment for the monthly obligation. And 4 times a year, the bill will contain one more bill segment (for a total of 3) for the quarterly obligation.

In sum, the account's bill cycle controls when the system attempts to create a bill for the account's obligations. Whether an obligation contributes a bill segment to the bill is a complicated subject. The topics in this section describe how the system knows it's time to create a bill segment for an obligation.

**Important!** An account's bill cycle should attempt to create bill segments at least as often as the shortest obligation duration. For example, if an account has both monthly and quarterly obligations, the account should be placed on a monthly bill cycle. Refer to [How Does An Account Get Its Bill Cycle?](#) for more information.

### Contents

- [Ways To Control The Start Date Of A Bill Segment](#)
- [Ways To Control The End Date Of A Bill Segment](#)
- [Preventing Short Bill Segments](#)

## Ways To Control The Start Date Of A Bill Segment

Bill segments produced for an obligation have two time periods:

- The bill segment period. The bill segment period defines the entire period of time covered by a bill segment's charges.
- The billable period. The billable period defines the period of time used to calculate the number of days for daily charges.

The billable period almost always starts one day after the bill segment period. The billable period always ends on the bill segment's end date. For example, a bill segment period that spans 5-Jan-2002 through 6-Feb-2002 will almost always have a corresponding billable period of 6-Jan-2002 through 6-Feb-2002. The reason that the start dates don't match is because a bill segment's start date equals the end date of the prior bill segment (i.e., the start date was already counted in the previous bill segment's period and we don't want to count it twice).

## Ways To Control The End Date Of A Bill Segment

The following points describe the different methods that can be used to define the end date on an obligation's bill segments:

- **Bill Period Schedule.** Obligations may have the end date of their bill segments defined on the user-maintained bill period calendar. This option is used when bill segments must fall on strict calendar boundaries (e.g., quarterly bills that end on the last day of the quarter).
- **Anniversary.** In addition to the Bill Period Calendar method, obligations may have their bill segment end date based on the obligation start date. For example, if an obligation started on the 16<sup>th</sup> of some month, the ongoing bill segments will start on roughly the 16<sup>th</sup> of each month.
- **ASAP.** For obligations that use neither the Bill Period Calendar nor Anniversary methods, the system assumes the end date of a bill segment is the current date.
- **Billable Charge.** For non-metered obligations that exist to levy billable charges, the start AND end dates are defined on the billable charge information that is entered by a user or interfaced from an external system.

The topics in this section discuss each of the above methods.

### Contents

[Using The Bill Period Schedule Method](#)

[Using The Anniversary Method](#)

[Using Billable Charges](#)

### Using The Bill Period Schedule Method

The bill period schedule method causes the bill segment end date to be calculated using a bill period schedule. The bill period's schedule defines when bill segments are produced and their respective end dates. This method is used when the bill segment needs to fall on strict calendar boundaries (e.g., quarterly bills that end on the last day of the quarter).

The obligation's obligation type indicates if the obligation uses this method. If so, the obligation type also specifies the bill period whose schedule defines the end dates.

**Future and Past End Dates.** It's important to be aware that a bill period schedule can be used to generate end dates that are in the future or in the past.

### Using The Anniversary Method

The anniversary date method causes the bill segment end date to be calculated based on the first day of service. For example, if an obligation started on the 16<sup>th</sup> of some month, the ongoing bill segments will start on roughly the 16<sup>th</sup> of each month. When using this method, you also define the applicable billing frequency (i.e., monthly, bi-monthly, quarterly, etc.).

The following table contains the bill periods for an obligation's starting on 23-Feb-99 that uses a monthly frequency.

Start Date	End Date	Billing Days
23-Feb-99	25-Mar-99	30
25-Mar-99	25-Apr-99	31

25-Apr-99	25-May-99	30
25-May-99	25-Jun-99	31
25-Jun-99	25-Jul-99	30
25-Jul-99	25-Aug-99	31
25-Aug-99	24-Sep-99	30
24-Sep-99	24-Oct-99	30
24-Oct-99	24-Nov-99	31
24-Nov-99	24-Dec-99	30
24-Dec-99	24-Jan-00	31
24-Jan-00	23-Feb-00	30
23-Feb-00	25-Mar-00	31

**Future and Past End Dates.** It's important to be aware that anniversary billing can generate end dates that are in the future or in the past. The future-option generates the next anniversary date after the current date. The past-option generates the anniversary date that immediately precedes the current date.

The obligation's obligation type indicates if the obligation uses this method. If so, the obligation type also specifies the billing frequency that determines the amount of time between bill segments.

For more information about the obligation type attributes that control this method, refer to [Setting Up Obligation Types](#).

### Using Billable Charges

Bill segments for billable charge obligations have both their start and end dates defined on the respective billable charge.

For more information about billing billable charges, refer to [How To Create An Ad-hoc Bill](#). For more information about creating a billable charge, refer to [Maintaining Billable Charges](#). For more information about interfacing a billable charge from an external system, refer to [Uploading Billable Charges](#).

The obligation type indicates if the obligation uses this method.

For more information about the obligation type attributes that control this method, refer to [Setting Up Obligation Types](#).

## Preventing Short Bill Segments

Every billable obligation type contains an attribute defining the minimum number of days on a bill segment. Whenever the system attempts to create a bill segment other than the final bill segment, it checks if the number of days is at least as great as the minimum. If not, the bill segment will not be created as part of this bill run. Rather, the system waits until the number of days in the bill segment is at least as large as the minimum. This is true for all methods of bill duration calculation.

## Prorating Charges When a Rate is Applied

It is possible for some of the prices that appear on a bill segment to change during the course of the bill period.

Refer to [Effective Dates & Proration](#) for more information about how charges are prorated during rate application.

## Batch Billing

If your implementation uses cyclical billing, most bills will be created by the batch bill process (known by the batch control ID of **BILLING**) and require no human intervention before they can be finalized and sent to a taxpayer. The others will be created by users on-line / real time. This section discusses several important concepts associated with batch billing.

### Contents

- [Window Billing And The Bill Cycle Schedule](#)
- [Confirming A Batch Of Bills Before Completing Them](#)
- [Canceling A Batch Of Bills After They're Complete](#)
- [Reopening A Batch Of Bills After They're Complete](#)
- [Fixing Errors Detected In Batch Billing](#)
- [Completing Pending Bills](#)

## Window Billing And The Bill Cycle Schedule

Refer to [The Cyclical Billing Process & Window Billing](#) for more information about how an account's bill cycle dictates when a taxpayer is billed.

## Confirming A Batch Of Bills Before Completing Them

If you're implementing new rates or if something unusual is being introduced that affects billing and your implementation is using the cyclical billing process, you may want to turn off the bill cycle schedule's Freeze and Complete switch. If this switch is off, the system creates bills, it just doesn't freeze and complete them. You can then review the entire batch of bills to make sure they're clean.

- If you find the bills are wrong, correct the source of the error and rerun the bill cycle. The system will remove all incomplete bills and then reproduce them using the corrected information in the system.

- If you find the bills are correct, turn the Freeze and Complete switch on and rerun the cycle. The system will remove all incomplete bills and then reproduce them. Assuming nothing changed, the bill amounts will be the same.

Refer to [Setting Up Bill Cycles](#) for more information.

### Canceling A Batch Of Bills After They're Complete

If you need to cancel an entire batch of bills because they were created using faulty data (e.g., the wrong tax rate was defined in the rate), you can. A background process called [MASSCNCL](#) exists for this purpose.

This background process will cancel all the frozen bill segments for the latest run of a given bill cycle's schedule. Optionally, you can cancel bills for a given bill date within the bill cycle's schedule. The cancel reason used on the bill segment is the one marked as the bill cancel reason for mass cancel. Refer to [Setting Up Bill \(Segment\) Cancellation Reasons](#) for more information.

When the cycle is billed again, new bill segments will be created, and the original bill segments and the cancellations will automatically be hidden from the taxpayers.

### Reopening A Batch Of Bills After They're Complete

If you need to reopen an entire batch of bills because they were completed prematurely, for instance if appraisal data did not get updated before billing ran, you can. A background process called [MASSROBL](#) exists for this purpose.

This background process will reopen all the bills for a given bill date for the latest run of a given bill cycle's schedule.

### Fixing Errors Detected In Batch Billing

If an "error" bill segment is created by the batch billing process, the system attempts to fix the offending segment at the next cyclical billing run by regenerating it using the information that exists at that time. Therefore, if the cause of the error is fixed during the day, the system will automatically regenerate the bill segment when batch billing next runs; you don't have to manually correct each bill. And, once a bill is error-free, it will be completed and sent to the taxpayer.

**Important!** Automatic regeneration only works during the account's bill window. If an "error" bill segment is not successfully regenerated on the last night of the account's bill cycle, it will remain in error until the next time the account's cycle runs (unless a user corrects it real time). At that time, the system will generate another bill error indicating the prior bill segment is in error (and then there'll be two bill segments in error).

### Completing Pending Bills

If you need to complete pending bills created by the batch billing process, you can. A background process called [C1-BLCMP](#), exists for this purpose.



This background process will complete all pending bills on a given bill date for a given bill cycle's schedule. This process does not delete and regenerate freezable bill segments linked to pending bills.

Refer to [Bill Errors](#) for more information.

## Billing Financial Transaction Considerations

The topics in this section provide information about the financial impact of a bill segment.

### Contents

[Billing - Current Balance versus Payoff Balance](#)  
[The Source Of GL Accounts On A Bill Segment's Financial Transaction](#)

### Billing - Current Balance versus Payoff Balance

A bill segment's financial transaction affects an obligation's payoff balance and/or current balance. In this section, we describe these two balances.

**Warning!** If you do not understand the difference between payoff balance and current balance, refer to Current Amount versus Payoff Amount.

### Contents

[When Current Balance Equals Payoff Balance](#)  
[When Current Balance Differs From Payoff Balance](#)  
[Bill Segment Type Controls Which Balance\(s\) Are Affected](#)

#### When Current Balance Equals Payoff Balance

For most obligations, payoff balance and current balance are always the same (or in colloquial speech – the amount the taxpayer thinks they owe equals what they really owe). Let's run through a typical example. The values in the payoff balance and current balance columns reflect the amount due after the financial transaction has been applied (i.e., the running balance):

Date	Financial Transaction	Payoff Balance	Current Balance
1-Jan-99	Bill: \$125	125	125
15-Jan-99	Payment: \$150	-25	-25
2-Feb-99	Bill: \$175	150	150
14-Feb-99	Payment: \$150	0	0
3-Mar-99	Bill: \$200	200	200
15-Mar-99	Payment: \$150	50	50
2-Apr-99	Bill: \$225	275	275

As you can see, payoff balance and current balance are always in sync.

### When Current Balance Differs From Payoff Balance

For some obligations, payoff balance and current balance differ (or in colloquial speech - the amount the taxpayer thinks they owe differs from what they would owe if they wanted to pay off their account).

**Note.** The base product does not currently support billable obligations types where the current balance would differ from payoff balance as described above.

### Bill Segment Type Controls Which Balance(s) Are Affected

Every bill segment references a bill segment type (the bill segment type comes from its obligation's obligation type). The bill segment type controls how payoff balance and current balance are affected by the bill segment amount. It also controls the algorithm used by the system to calculate the bill segment's bill lines.

Refer to [Defining Bill Segment Types](#) for more information about how bill segment type affects how a bill segment is produced and how its financial transaction is generated.

### The Source Of GL Accounts On A Bill Segment's Financial Transaction

A bill segment's financial transaction also contains the double-sided accounting entry that defines how the bill segment affects the general ledger.

Refer to [The Source Of GL Accounts On Financial Transactions](#) for a description of where the system extracts the distribution codes used to construct the GL accounts.

## The Source Of Bill Routing Information

When a bill is completed, the system creates a [bill routing](#) for each person linked to the account who receives a copy of the bill. The bill routing record contains the information that controls how, where and to whom a bill is sent.

The following points describe how this works:

- Each person who receives a copy of an account's bill is listed on the [account's person information](#).
- A [bill routing](#) is created for each such person when the bill is completed.
- The information that appears on the bill routing record is controlled by the **bill route type** specified on the respective account / person information:
  - If the bill route type indicates the person's bills are routed via [fax](#), the "address line 1" address constituent is populated with the person's fax number. The system knows which of a [person's phone numbers](#) is a fax number by the phone type. If the person has multiple fax numbers, one is selected at random.
  - If the bill route type indicates the person's bills are routed via [email](#), the "address line 1" address constituent is populated with the person's [Email address](#).

- If the bill route type indicates the person's bills are routed via the postal service, the address constituents are populated with the address specified on the account / person's bill address source. Note, if the person has a seasonal address effective on the business date, the seasonal address will be used regardless of the value of **bill address source**.

Refer to [Printing Bills](#) and [How To Reprint A Bill \(For The Original Recipients or For Someone New\)](#) for more information. Refer to [Account – Person Information](#) for information regarding how to control who receives a copy of a bill, where the bill is sent, and how the bill is formatted.

## Bill Messages

The topics in this section describe how messages are linked to a bill and bill segment.

### Contents

[The Source Of Bill Messages](#)  
[Substituting Field Values Into A Message](#)

### The Source Of Bill Messages

When a bill is completed, the system sweeps bill messages from the following sources onto the bill.

**Bill or Bill Segment Messages.** Bill messages will be linked either to the bill or to one of its bill segments depending on the source of the bill message code. In other words, messages associated with an obligation (directly or indirectly) will be linked to the bill segment; messages associated with an account (directly or indirectly) will be linked to the bill.

- All permanent and temporary bill messages linked to an account are linked to the bill when it is completed. Refer to [Account – Bill Messages](#) for more information. Note well that all temporary bill messages are removed from the account when they are swept onto a bill (a temporary message is swept onto the next bill produced for the account).
- The system checks if the bill's account's account type has bill messages. If so, it links all such messages that are effective on the bill date to the bill when it is completed. Refer to [Setting Up Account Type Bill Messages](#).
- All permanent and temporary bill messages linked to obligations that contributed bill segments to the bill are linked to their respective bill segment when the bill is completed. Refer to [Obligation – Miscellaneous](#) for more information. Note well that all temporary bill messages are removed from the obligation when they are swept onto a bill segment (a temporary message is swept onto the next bill segment produced for the obligation).
- The system checks if each bill segment's rate has bill messages. If so, it links all such messages that are effective on the bill segment's start date to the bill segment when the bill is completed. Refer to [Rate Schedule – Bill Messages](#).

In addition, a user may manually add an [ad hoc message to a bill](#).

And finally, you can develop your own background processes and algorithms that add bill messages to accounts, obligations, bills and/or bill segments. Refer to [Substituting Field Values Into A Bill Message](#) for examples.

## Substituting Field Values Into A Message

Many bill messages contain static text (i.e., the message is the same on every bill). However, the system supports messages whose contents are dynamic. For example, consider the bill message ***Your 2007 taxes were reduced due to your homestead deduction of \$6,000.*** This message contains two substitution values (the year and the amount) as is therefore considered dynamic.

Dynamic messages can be implemented as follows:

- Create a bill message code whose Message on Bill contains substitution values. For example, the bill message code to produce the message illustrated above would contain the following Message on Bill - ***Your %1 taxes were reduced due to your %2 of %3.***
- Use either of the following methods to link the message code and its substitution parameters to the bill:
  - Create a background process or algorithm to insert a permanent or temporary bill message on the appropriate accounts. Then, when an account's bill is next completed, the system will sweep the message (and its substitution parameters onto the bill).

In addition to inserting the appropriate bill message code, your background process / algorithm must also insert the appropriate substitution values. The name of the table in which account messages are inserted is [CI\\_ACCT\\_MSG](#). The name of the table in which a message's substitution values are inserted is [CI\\_ACCT\\_MSG\\_PRM](#) (you will insert one row for each substitution field).

- Rather than adding the dynamic message to account (and then letting the bill completion logic transfer the account message to the bill), you could construct a bill completion algorithm that adds the message code and substitution values on the bill (during bill completion). The name of the table in which bill messages are inserted is [CI\\_BILL\\_MSGS](#). The name of the table in which a message's substitution values are substituted is [CI\\_BILL\\_MSG\\_PRM](#) (you will insert one row for each substitution field).

## A Bill May Affect More Than Just Taxpayer Balances

The topics in this section provide information about obscure things that may happen when a bill is created, frozen, or canceled.

### FT Freeze Repercussions

Refer to [Obscure Things That Can Happen](#) for more information about things that can happen when an FT is frozen (and FT's get frozen during billing).

## Using Billable Charges for Pass Through / Convergent Billing

The term "pass through" billing is used to describe the practice of receiving charges calculated by third parties and presenting them on the taxpayer's bill along with your own charges. "Pass through" billing is implemented in the system using [Billable Charges](#).

The following points provide information to help you decide the most appropriate way to implement "pass through" billing given your specific requirements:

- Taxpayers with pass through charges will need a separate obligation to hold the pass through charges. We refer to this type of obligation as a “billable charge” obligation.

**Note.** An Obligation’s obligation type controls whether an obligation can have billable charges linked to it. Specifically, the obligation type must have a “special role” of **Billable Charge**.

- A billable charge obligation holds the billable charges until the taxpayer is next billed. At that time, a separate bill segment will be created for each unbilled billable charge linked to the billable charge obligation.
- While it is not required, we recommend creating a separate billable charge obligation for each type of pass through charge.
- You can interface billable charges using the [Billable Charge Interface](#). The interfaced charges can consist of any of the following:
  - Pre-calculated bill lines that will be presented “as is” to the taxpayer.
  - Rate quantities that are used by the system to calculate the charges on behalf of the third party.
- The bill lines on a billable charge can fit into any of the following categories:
  - Memo-only (i.e., don’t affect the general ledger). A bill line that is memo-only contains information that is purely informational.
  - Show on the taxpayer bill. It is possible to create bill lines that affect the general ledger, but are not shown on the taxpayer’s bill. This is an unusual practice, but it happens.
  - Summary / detail. Each bill line has an indication of whether it is a summary or detail line. This only impacts bill presentation.

**Note.** The above indicators can be defaulted onto a billable charge line by using [Billable Charge Line Types](#) when you interface the lines into the system.

- Characteristics (i.e., user-defined fields) can be associated with billable charge lines. You might populate this information if you are interfaced information that may be useful during bill presentation or for reporting purposes.
- If rate quantities are specified on a billable charge, they are saved on the bill segment created when the billable charge is billed.
- Cancel / rebill for billable charges is quite different than for normal bill segments. A billable charge bill segment can be cancelled, and this will reverse the financial effects of the billable charge. But without new information from the source, there is no way to rebill the taxpayer. Therefore, if the original charges were incorrect, the source system would send both a reversal of the charges and a newly revised set of information. These could be passed as two separate billable charges or they could be combined on a single billable charge.
- For most other functionality, the billable charge obligation supports the same functionality as normal obligations. This includes payment distribution, collections, current & payoff balances, etc.

For more information about billing billable charges, refer to [How To Create An Ad-hoc Bill](#). For more information about creating a billable charge, refer to [Maintaining Billable Charges](#). For more information about interfacing a billable charge from an external system, refer to [Uploading Billable Charges](#).

## Printing Bills

The contents of this section describe the technical implementation of both an online and batch bill production.

### Contents

- [Bill Routings Are Created For Each Recipient Of A Bill](#)
- [Bill Route Types Control The Information Merged Onto Bills](#)
- [Technical Implementation Of Online Bill Images](#)
- [Technical Implementation Of Printing Bills In Batch](#)
- [Reproducing The Bill Print Flat File](#)
- [How To Reprint A Specific Bill](#)
- [Who Gets A Copy Of A Bill?](#)
- [Final Bills and Bill Print](#)

### Bill Routings Are Created For Each Recipient Of A Bill

Refer to [The Source Of Bill Routing Information](#) for a description of how the system constructs the information used to route a bill to one or more recipients.

### Bill Route Types Control The Information Merged Onto Bills

Every bill routing record references a [Bill Route Type](#). The bill route type controls the following:

- It contains an algorithm that is responsible for extracting the information merged onto your bills. Algorithms of this type are called under the following scenarios:
  - The [background process](#) that interacts with your bill print software calls these algorithms.
  - If your bill print software has the ability to construct a real-time image of a bill, you can plug-in an **Online Bill Display** algorithm on the [Installation Record](#). This type of algorithm will call the extract algorithm for an appropriate bill route type to extract the information that is merged onto the bill. Refer to [Technical Implementation Of Online Bill Image](#) for the details.
- It contains the ID of the background process responsible for interacting with your bill print software in batch.

### Technical Implementation Of Online Bill Images

Users can view an image of any bill that is sent to a taxpayer on [Bill - Main](#) if you set up the following:

- Plug-in an **Online Bill Display** construction algorithm on the [Installation Record](#).
- Plug-in the appropriate bill extract algorithm on each [Bill Route Type](#).

The system provides algorithms that interact with bill print software that renders an image of the bill in a PDF. The following points describe what takes place when clicking **Display Bill** when these sample algorithms are used.

- The sample **Online Bill Display** algorithm [ONLN-BL-DSP](#) is executed.
- This algorithm calls the bill extract algorithm for an appropriate bill route type (as determined by the algorithm). The sample bill extract algorithm [BLEX-EX](#) constructs the information that appears on the bill and returns it to the **Online Bill Display** algorithm. This algorithm, in turn, passes it to your bill print software.
- Your bill print software renders an image of the bill in a PDF and returns it to the **Online Bill Display** algorithm.
- And finally, the system displays the PDF in a separate browser session. Note that the client must have Adobe Acrobat Reader installed to view PDF files.

## Technical Implementation Of Printing Bills In Batch

The batch process that extracts bill information reads all bill routing records in a given run number that are marked with its batch control ID

The base package example of this process (known by the batch control ID of [POSTROUT](#)) simply calls the extract algorithm on the routing record's route type to format the information placed onto the flat file. Refer to [Bill Route Types Control The Information On Bills](#) for more information.

**If your software doesn't support online bill images.** The algorithm that formats bill extract records that's plugged in on [bill route type](#) serves two purposes: 1) it interacts with the online bill display algorithm plugged into the installation record to support online images of a bill, and 2) it interacts with the background process to support printing your bills in batch. If your bill print software does not support the rendering of bill images real time, there is no need to create an extract algorithm. Rather, you should simply develop your own download process that works with your bill print software to print bills in batch (and then specify this batch process on your bill route type).

## Reproducing The Bill Print Flat File

You can reproduce the flat file at any time. Simply request the download process and specify the run number associated with the historic run.

## How To Reprint A Specific Bill

Refer to [How To Reprint A Bill \(For The Original Recipients or For Someone New\)](#) for instructions describing how to reprint a specific bill.

## Who Gets A Copy Of A Bill?

A copy of a bill is sent to every individual / business specified in the [bill's routing details](#).

There are two ways in which a bill routing detail can be created:

- When a bill is completed, the system creates a routing detail for every person linked to the account that wants to receive a copy of the bill (as specified on [Account – Person Information](#)). Refer to [The Source Of Bill Routing Information](#) for more information.
- After a bill is completed, you may insert a bill routing record. Refer to [How To Reprint A Bill \(For The Original Recipients or For Someone New\)](#) for more information.

## Final Bills and Bill Print

If a bill is produced for an account where all of the account's obligations are **stopped**, **closed** or **cancelled**; the bill is flagged as being the final bill on the flat file produced by the postal routing process.

If a bill is considered to be a final bill and the amount owing for the entire account is less than the final bill threshold amount on the installation record, the bill will be skipped (i.e., it won't appear on the flat file produced by the postal routing process). Note, this logic also suppresses bills from being produced when a payment is received AFTER the final bill and the account's balance falls beneath the installation record's final bill threshold amount. Refer to [Installation Options - Billing](#) for more information.

## Idiosyncratic Manual Bill Cancellation

If a bill's account is associated with an account type that has a **Cancel Bill** algorithm, the system invokes this algorithm before the bill is displayed to determine if the bill is "cancellable". If the algorithm indicates the bill is cancellable, a button appears on [Bill - Main](#). When clicked, the **Cancel Bill** algorithm is invoked again to cancel the bill. This functionality is meant to support unique cancellation needs required by some implementations; the sample base-package algorithm is empty.

## Maintaining Bills

---

A bill is used to communicate changes in a taxpayer's financial obligations to the taxpayer. The topics in this section describe how to maintain bills.

It's important to be aware that there are very few fields that are directly modifiable by a user. To modify most fields on a bill, you have to change source information (e.g., obligation, rate) and then regenerate the bill. For example, if you want to change a bill's amount, you must cancel or add bill segments; you cannot change the bill's amount by modifying the bill amount field. Refer to [How To](#) for step-by-step instructions that explain how to perform common bill maintenance functions.

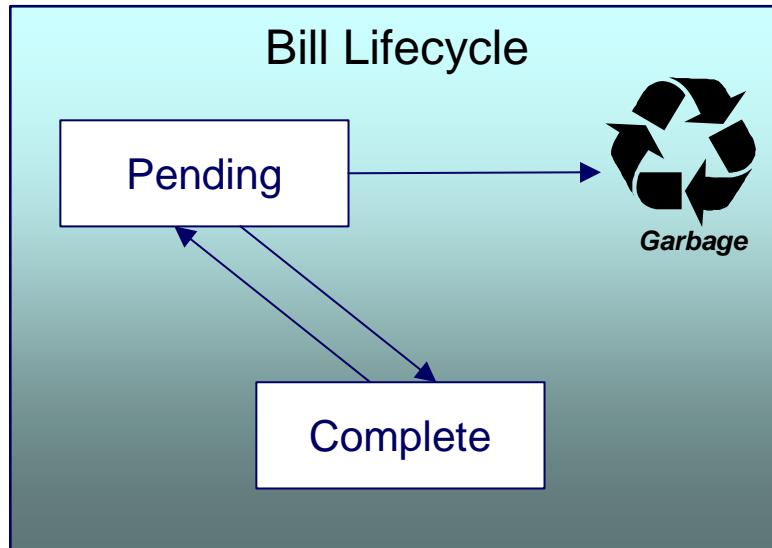
### Contents

- [Bill Lifecycle](#)
- [Bill - Main Information](#)
- [Bill - Bill Segments](#)
- [Bill - Bill Routings](#)
- [Bill - Bill Messages](#)
- [Bill - Characteristics](#)
- [How To](#)

## Bill Lifecycle

The following diagram shows the possible lifecycle of a bill.





**Warning!** This explanation only makes sense in the context of the page used to maintain bills. Refer to [Bill – Main Information](#) for the details.

A bill is initially saved in the **Pending** state. You may create one or more bill segments for the account's obligations at this point. Refer to [How To](#) for information about generating bill segments for the bill.

A bill becomes **Complete** when it is ready to be sent to the taxpayer. Completing a bill triggers many things to occur. Refer to the section below for information about what happens when a bill is completed.

When you **complete** a bill, several things may happen:

- Pre-completion algorithms associated with the account's [account type](#) are executed.
- The bill's due date is calculated. This is equal to the bill date plus the number of days defined on the account's account type. If the resultant date is not a workday, the due date is set to the next workday. Note, this due date can be overridden if an override algorithm exists on the account's account type.
- The bill's routing information is set up using [Account – Person Information](#).
- If the **Freeze At Bill Completion** [installation option](#) has been turned on, **freezable** bill segments and adjustments linked to the account are frozen. Note, only **freezable** adjustments whose [adjustment type](#) indicates **Freeze At Bill Completion** will be frozen at this time.
- Post completion algorithms associated with the account's obligations' [obligation types](#) are executed. The system executes these algorithms first in the order of the [billing processing sequence](#) on each obligation's obligation type then in the order of the algorithm's sequence.
- Bill completion algorithms associated with the account's [account type](#) are executed.
- Bill messages are amalgamated from various sources and linked to the bill. Refer to [The Source Of Bill Messages](#) for more information.
- Other financial transactions that have been frozen since the last bill and are marked to show on bill are linked to the new bill

- For open-item account types adjustments and corrections are linked
- For all other situations, payments, adjustments and corrections are linked
- All financial transactions that don't already have a user-defined aging date will be dated with the current date. In other words, they start aging from the date the bill is completed.
- If the account's account type indicates the account is an open-item taxpayer, a match event is created. The bill's FTs and the automatic payment's FTs are linked to it. Refer to [Payments and Match Events](#) for more information about match events.
- If the account pays, the bill is stamped with the date when the automatic payment is to be created. Refer to [The System Background Processes](#) for more information about **APAYCRET**, the background process that creates the automatic payment and **APAYDSFR**, the background process that distributes and freezes the automatic payment.
- If the account's account type indicates the account is an open-item taxpayer, the system will create a match event if the new charges are offset by other financial transactions. Refer to [Payments and Match Events](#) for more information about match events.
- The bill's status becomes **Complete**.
- Post bill completion algorithms associated with the account's [account type](#) are executed.

If the system cannot complete the bill (for whatever reason), the bill remains **Pending** and an error message is shown on the main bill page. After correcting the cause of the problem, attempt to complete the bill again.

A **complete** bill may be changed back to **pending** using the [Reopen](#) button on [Bill – Main Information](#). You would reopen a bill when

- Add more bill segments to a completed bill. Refer to the [How To](#) section for information about linking bill segments for a bill.
- Fine-tune the payments, adjustments, and corrections that were linked to a completed bill. Refer to the [How To](#) section for detailed instructions.

When you're happy with the bill, you can complete it again.

**Cannot reopen historical bills.** You may only reopen an account's most recent bill because recompleting the bill causes the ending-balance to change, and we don't want this to happen to historical bills.

**Automatic payments.** If an automatic payment was created when the bill was completed and it has already been interfaced to the financial institution, you cannot reopen the bill. If the automatic payment exists, but it has not yet been interfaced, the system will automatically cancel the payment when you reopen the bill.

You may [delete](#) a **Pending** bill from the database. You may not delete a pending bill if: a) there are frozen bill segments linked to the bill, or b) if financial transactions were linked to the bill (and this can only happen if the bill had been previously completed).

In addition to removing the bill, the system will also remove unfrozen bill segments linked to the bill.

## Bill - Main Information

Open **Financial, Bill, Main** to maintain core bill information.

The follow discussion simply describes the fields on this page. Refer to [How To](#) for a description of how to perform common bill maintenance functions.

### Description of Page

**Bill Info** contains a concatenation of important information about the bill (e.g., the bill date, its status, due date, ending balance, etc.).

**Bill ID** is the system-assigned unique identifier of the bill.

**Account ID** identifies the taxpayer who is responsible for the bill. The name of the main taxpayer on the account appears adjacent.

**Bill Status** is the bill's status. Refer to [Bill Lifecycle](#) for the potential values and how to handle a bill when it exists in a given state. Adjacent to the bill status is the **Display Bill** button, which will display an online image of the bill when clicked. Refer to [How To Display A Bill On-line](#) for more information.

**Note.** You can only use the **Display Bill** button if your system has been configured to display an on-line image; otherwise, a message indicating that the service is not available will appear. This option can only be configured by your technical staff.

**Due Date** is the date on which the bill is due. A bill's due date is calculated as follows:

- Add the due days specified on the account's account type to the bill date.
- The resulting date is the due date unless it's not a workday. In this case, the due date is set to the next workday (workdays are defined in the installation options tables).
- If the account's account type has an override due date calculation algorithm, the due date may be overridden. Refer to [Setting Up Account Types](#) for more information.

**Bill Date** is the business date that was used when the bill was completed.

**Create Date/Time** is the date and time on which the bill was originally created.

**Completion Date/Time** is the date and time on which the bill was completed.

If the taxpayer [pays automatically](#) and the automatic payment has not yet been created, a brief message appears highlighting the date and amount of the future automatic payment. In addition, a button appears called **Stop Autopay**, that, when clicked, causes the associated automatic payment to stop. You might click this button if the taxpayer indicates that they will send in a payment. After you click the **Stop Autopay** button, a message appears highlighting the stoppage. Refer to [How And When Are Automatic Payments Created](#) for more information about how to setup the system to defer the creation of automatic payments until the future automatic payment extract date.

In the **Message** area, a brief error message appears if there's a problem with the bill. The message area is suppressed if there are no problems with the bill.

Click the magnifying button to view the long explanation. The long explanation will provide information about the cause of the error (and how to fix it).

**Bill Summary** summarizes the financial impact of the bill. The information that appears differs depending on whether the account is a [balance-forward or open-item account](#). If the account is a balance-forward taxpayer, the following information appears:

- **Previous Period's Balance** is the balance from the account's last bill.
- **Total Payments** is the total amount of frozen or canceled payment segment financial transactions linked to this bill. Frozen payment segments appear as negative numbers (decreasing the amount owed by the taxpayer), while canceled payment segments appear as positive numbers (increasing the amount owed). You can drill down on this balance to view the financial transactions that contribute to this total. Refer to [How To Remove Unwanted Payments From A Completed Bill](#) for a description of how to perform common maintenance functions.
- **Total Adjustments** is the total amount of frozen or canceled adjustment financial transactions linked to this bill. You can drill down on the balance to view the financial transactions that contribute to this total. Refer to [How To Remove Unwanted Adjustments From A Completed Bill](#) for a description of how to perform common maintenance functions.
- **Total Bill Corrections** is the total amount of canceled and / or rebilled bill segment financial transactions linked to this bill. You can drill down on the balance to view the financial transactions that contribute to this total.
- **Total Current Billing Charges** is the total amount of frozen bill segment financial transactions linked to this bill. You can drill down on the balance to view the financial transactions that contribute to this total.
- **Ending Balance For This Period** is the sum of the five amounts above. It is the amount owed by the taxpayer as of the end of the bill period.

If the account is an [open-item taxpayer](#), the following information appears under **Bill Summary**:

- **New Charges** is the total amount of frozen bill segment financial transactions linked to this bill. You can drill down on this value to view the financial transactions that contribute to this total.
- **Adjustments** is the total amount of frozen or canceled adjustment financial transactions linked to this bill. You can drill down on this value to view the financial transactions that contribute to this total. Refer to [How To Remove Unwanted Adjustments From A Completed Bill](#) for a description of how to perform common maintenance functions.
- **Corrections** is the total amount of canceled and / or rebilled bill segment financial transactions linked to this bill. You can drill down on this value to view the financial transactions that contribute to this total.
- **Total** is the sum of the above amounts. You can drill down on this value to view the financial transactions that contribute to this total.

If the account is an [open-item taxpayer](#), the following information appears under **Match Summary** (note, this section is not displayed for balance-forward accounts):

- **X Balanced Item(s)** contains the count and total amount of financial transactions linked to this bill that are linked to **balanced** match events.
- **X Unbalanced Item(s)** contains the count and total amount of financial transactions linked to this bill that are linked to **unbalanced** match events.
- **X Disputed Item(s)** contains the count and total amount of financial transactions linked to this bill that are linked to **disputed** match events.

- X **Unmatched Item(s)** contains the count and total amount of financial transactions linked to this bill that are not linked to any match event.

**Total Changes After Completion** only appears if bill segments that appeared on the original bill have been subsequently [canceled or rebilled](#). The amount displayed is the net effect of all cancels and rebills. Note, you can use the **Remarks** column in the bill segments grid (to the right) to see which bill segments have been cancelled / rebilled.

If 25 or fewer bill segments are linked to this bill, the grid in the lower right portion of the page contains one row for every bill segment linked to the bill.

The following points describe each column:

- The **Bill Segment** column contains a concatenation of each bill segment's division, obligation type, status and bill period. The division and obligation type come from the bill segment's obligation.
- The **Current Amount** column contains the bill segment's effect on the obligation's current balance.
- The **Status** column shows the bill segment's status. Refer to [Bill Segment Lifecycle](#) for the possible values.
- The **Remarks** highlight special situations. The following remarks may appear:
  - **Rebill after completion** appears if the bill segment is a rebill that was created after the bill was sent to the taxpayer. The financial impact of this bill segment appears as a "bill correction" on the next bill produced for the taxpayer.
  - **Rebill prior completion** appears if the bill segment is a rebill that was created before the bill was sent to the taxpayer.
  - **Canceled after completion** appears if the bill segment was canceled after the bill was sent to the taxpayer. The financial impact of this bill segment appears as a "bill correction" on the next bill produced for the taxpayer.
  - **Canceled prior completion** appears if the bill segment was canceled before the bill was sent to the taxpayer.
  - If the bill segment's status is **Error**, this column also contains the error message.

If more than 25 bill segments are linked to this bill, the grid in the lower right portion of the page contains a summary of the various bill segments linked to the bill:

The following points describe each column:

- The **Status** column shows the bill segment's status. Refer to [Bill Segment Lifecycle](#) for the possible values. Click on the hyperlink to transfer to the Bill Segments tab where the associated bill segments can be viewed.
- The **Total Bill Segments** column contains the number of bill segments linked to the bill with this **Status**.
- The **Total Amount** column contains the sum of the current amount of these bill segments.

At the bottom of the scroll is shown the **Total Generated Charge**. This represents the total amount of the bill segments. This amount may differ from the Total Current Billing Charges when:

- The bill is not **Complete**. A bill segment's financial effect is not shown on the bill until the bill is completed.

- A bill segment has been canceled / rebilled since the bill was completed.

The topics that follow describe each of the actions that appear in the **Bill Segment Action** and **Bill Action** areas. Refer to [How To](#) for a description of typical business processes that use these buttons.

## Contents

[Generate](#)  
[Freeze](#)  
[Cancel Frozen](#)  
[Freeze / Complete](#)  
[Complete](#)  
[Delete](#)  
[Reopen](#)  
[Cancel Bill](#)

## Generate

The **Generate** button causes a bill segment to be created for each billable obligation linked to the account. The system generates bill segments in the order of the billing processing sequence on each obligation's obligation type.

This button is enabled when you are in add mode (i.e., you are not displaying an existing bill) and you have selected an **Account ID**.

When clicked, the **Generate** window opens.

You must specify the following parameters in the **Generate** window to generate the bill segments:

- **Cut-off Date** is the last possible day of the bill segment's bill period. The current date defaults when the window opens. Refer to [Ways To Control The End Date Of A Bill Segment](#) for more information.
- **Accounting Date** is used to define the financial period to which the new bill segment's financial transaction will be booked. The current date defaults when the window opens.

After specifying the parameters, click **Calculate** to create a new bill and new bill segments for the account. You see a summary of the bill segments in the lower right portion of the page.

## Freeze

Clicking **Freeze** causes all **freezable** bill segments to become **frozen**. Refer to [Bill Segment Lifecycle](#) for more information about freezing.

This button is enabled if:

- the **Freeze At Will Bill Segment Freeze Option** on the [installation option](#) has been selected AND
- **freezable** bill segments exist AND
- there are no **error**, **incomplete** or **pending cancel** bill segments.

**Freezing a day or more after generation.** If, during freezing, bill segments in **closed** accounting periods are detected, a pop-up appears in which you can specify a new accounting date; this date is updated onto the offending bill segments. This only happens if the accounting period closes before you freeze the bill segments (the accounting date is stamped on a bill segment when it's generated).

If, after freezing, you're ready to send the bill to the taxpayer click [Complete](#) to finalize the bill.

**If problems are detected after freezing.** A bill segment may not be changed after it is **frozen**. All subsequent changes must occur by canceling the frozen bill segment and creating a new bill segment. Refer to [How To Cancel A Bill Segment](#) and [How To Cancel / Rebill A Bill Segment](#) for more information.

## Cancel Frozen

Clicking **Cancel Frozen** causes all **frozen** bill segments to become **canceled**. Refer to [Bill Segment Lifecycle](#) for more information about cancellation. This button is disabled if the bill is written off.

**Pending cancels are not affected.** Clicking **Cancel Frozen** does not affect bill segments that are **pending cancel**. To make a **pending cancel** bill segment **canceled**, transfer to [Bill Segment – Main](#) and click [Cancel](#).

If you need to cancel a specific bill segment, either transfer to the next tab and cancel the bill segment in question or follow the instructions under [How To Cancel A Bill Segment](#).

This button is enabled if **frozen** bill segments are linked to the bill.

You must specify the following parameters in order to cancel the frozen bill segments:

- **Cancel Reason** defines why you are performing the cancellation.
- **Accounting Date** defines the financial period to which the canceled bill segments' financial transactions are booked. The current date defaults when the window opens.

After specifying the parameters, click **Calculate** to cancel the frozen bill segments.

**There is no Undo.** Unlike cancellations performed on [Bill Segment – Main](#), there is no Undo action. This transaction moves the pending cancel bill segments to the cancel state immediately; whereas the bill segment transaction lets you examine the cancellation before you commit it. If you cancelled bill segments by mistake, you must generate and freeze new bill segments.

If the bill is **pending**, the cancellation causes the bill segments to be suppressed on the version of the bill sent to the taxpayer (but they remain in the database for audit and financial reporting purposes).

If the bill is **complete** and you do not [Reopen](#) the bill, the financial impact of the canceled bill segments appears on the next bill created for the account (under Bill Corrections).



If the bill is **complete** you may be able to [Reopen](#) the bill and then [Complete](#) it. By doing this, you can suppress a **frozen** bill segment on a previously completed bill. This means that if you catch a problem after completion you don't necessarily have to show the problem to the taxpayer. Rather, cancel the problem, reopen and then recomplete the bill (when you recomplete the bill the system marks the bill segment to be suppressed on the version sent to the taxpayer because its cancellation appears on the same bill as the original charges).

## Freeze / Complete

The **Freeze / Complete** button performs a variety of functions:

- all **freezable** bill segments (including rebills) become **frozen**
- all **pending cancel** bill segments become **canceled**
- all **freezable** adjustments whose [adjustment type](#) indicates **Freeze At Bill Completion** become **frozen**
- the bill is finalized and marked for transmission to the taxpayer

Refer to [Bill Lifecycle](#) for information about what happens during bill completion. Refer to [Bill Segment Lifecycle](#) for more information about freezing.

This button is enabled if:

- the **Freeze At Bill Completion** Bill Segment Freeze Option on the [installation option](#) has been selected AND
- the bill is **pending** AND
- there are no bill segments or there are bill segments and all are **freezable**, **pending cancel**, **frozen** or **canceled** AND
- there are no **error** or **incomplete** bill segments.

If the **User May Override Bill Date** option has been enabled on the [Installation Record](#), you may override the **Bill Date** prior to completion. Otherwise, the **Bill Date** is the current date and cannot be changed. The current date defaults when the window opens.

**Freezing a day or more after generation.** If, during freezing, bill segments in **closed** accounting periods are detected, a pop-up appears in which you can specify a new accounting date; this date is updated onto the offending bill segments. This will only happen if the accounting period closes before you freeze the bill segments (the accounting date is stamped on a bill segment when it's initially generated).

After specifying the parameter, click **Calculate** to freeze and complete the bill.

## Complete

The **Complete** button causes a bill to be finalized and marked for transmission to the taxpayer. Refer to [Bill Lifecycle](#) for information about what happens during bill completion.

This button is enabled if:

- the **Freeze At Will** Bill Segment Freeze Option on the [installation option](#) has been selected AND
- the bill is **pending** AND



- there are no bill segments or there are bill segments and all are **frozen** or **canceled**.

If the **User May Override Bill Date** option has been enabled on the [Installation Record](#), you may override the **Bill Date** prior to completion. Otherwise, the **Bill Date** is the current date and cannot be changed. The current date defaults when the window opens.

After specifying the above parameters, click **Calculate** to complete the bill.

## Delete

The **Delete** button deletes a bill and its bill segments.

This button is enabled if:

- the bill is **pending** AND
- all bill segments are **incomplete**, **freezable** or in **error**.

## Reopen

The **Reopen** button causes a bill to be reopened. Refer to [Bill Lifecycle](#) for information about why you would reopen a bill.

This button is enabled if:

- the bill is **complete** AND
- the bill is not written off AND
- this is the most recent bill produced for the account AND
- reopening hasn't been prohibited for the bill. Refer to [Bill Lifecycle](#) for information about conditions that can prevent a bill from reopening.

After reopening a bill, it returns to the **pending** state and you can make changes to the account's financial transactions (e.g., add payments, adjustments, and bill segments). After which, you click [Complete](#) or [Freeze / Complete](#) to finalize the bill. Only one of these buttons is shown – the specific one depends on how your organization has set the **Bill Segment Freeze Option** on the [installation record](#).

## Cancel Bill

Refer to [Idiosyncratic Manual Bill Cancellation](#) for a description of when the **Cancel Bill** button appears and what happens if the button is clicked.

## Bill - Bill Segments

You can use this page to view all or selected bill segments linked to a bill. In addition, you can also perform certain maintenance functions on this page (see the details below).

Open **Financial, Bill, Bill Segments** to view this information.

**Note.** If the bill has more than 25 bill segments and you don't use the hyperlinks in the bill segment grid on the Main tab to drill over to this information, the search criteria are intentionally left blank in order to avoid retrieving all bill segments (with the resultant slow response times). You must therefore use the **Obligation Filter** and the **Status Filter** to define the type of bill segments that should be retrieved. See the [Description of page](#) below for more information about this page's search criteria.

The [Description of page](#) that appears below simply describes the fields on this page. Refer to [How To](#) for a description of how to perform common maintenance functions.

### Description of page

**Bill Info** contains a concatenation of important information about the bill (e.g., the bill date, its status, due date, ending balance, etc.).

**Bill ID** is the system-assigned unique identifier of the bill.

**Note.** If the bill has more than 25 bill segments and you don't use the hyperlinks in the bill segment grid on the Main tab to drill over to this page, the **Filters** are intentionally left blank in order to avoid retrieving all bill segments (with the resultant slow response times). You must therefore use the **Obligation Filter** and the **Status Filter** to define the type of bill segments that should be retrieved.

The area beneath **Bill Info** provides you with options that control which bill segments appear in the grid. The following points describe the various options:

- Use the **Obligation Filter** to define the types of obligations whose bill segments appear in the grid. The following options are available:
  - **All.** Use this option if you do not wish to restrict bill segments based on obligation attributes.
  - **Obligation Type.** Use this option to restrict bill segments to those whose obligations are linked to a given **division** and **obligation type**.
- Use **Status Filter** to restrict the bill segments based on their status. The following options are available:
  - **All.** This option shows all bill segments regardless of status.
  - Refer to [Bill Segment Lifecycle](#) for the various status values.

Don't forget to click the search button after changing the filters.

The **Select All** / **Clear All** buttons are used to select bill segments if you plan on issuing any of the mass update actions at the bottom of the page. These buttons are enabled if either of the following conditions is true:

- All bill segments that appear in the grid are in the **Error**, **Freezable** and/or **Incomplete** states.
- All bill segments that appear in the grid are in the **Frozen** state

Refer to the description of the "mass update" actions below for more information.

**100 bill segments at a time.** Clicking **Select All** selects the first 100 bill segments in the grid. If more than 100 bill segments exist, you must select them in batches.

The grid that follows contains the bill segments that match your search criteria. The following information appears in the grid:

- **Select box.** See the mass update description immediately above for conditions under which this checkbox can be used to select bill segments for mass update actions.
- The **Bill Segment** column contains a concatenation of the bill segment's division, obligation type, status and bill period. The division and obligation type come from the bill segment's obligation. Click the hyperlink to transfer to [Bill Segment – Main Information](#). On this page you can perform maintenance functions (e.g., cancel/rebill, delete, cancel, etc., depending on the segment's status) on the bill segment in question.
- The **Current Amount** column contains the bill segment's effect on the obligation's current balance.
- The **Status** shows the bill segment's status. Refer to [Bill Segment Lifecycle](#) for the possible values.
- The **Remarks** column highlights special circumstances associated with the bill segment.
  - **Rebill after completion** appears if the bill segment is a rebill that was created after the bill was sent to the taxpayer. The financial impact of this bill segment appears as a "bill correction" on the next bill produced for the taxpayer.
  - **Rebill prior completion** appears if the bill segment is a rebill that was created before the bill was sent to the taxpayer.
  - **Canceled after completion** appears if the bill segment was canceled after the bill was sent to the taxpayer. The financial impact of this bill segment appears as a "bill correction" on the next bill produced for the taxpayer.
  - **Canceled prior completion** appears if the bill segment was canceled before the bill was sent to the taxpayer.
  - If the bill segment's status is **Error**, this column also contains the error message.
- **Rate Quantity** and **UOM** columns are not used.
- The **Bill Segment ID** is the system-assigned unique identifier of the bill segment.

This transaction has sophisticated logic that can be used to perform "mass updates" to the bill segments that appear in the grid (using the buttons at the bottom of the page). The following points describe these mass update actions:

### Contents

[Generate \(Bill - Bill Segments\)](#)  
[Freeze \(Bill - Bill Segments\)](#)  
[Delete \(Bill - Bill Segments\)](#)  
[Cancel/Rebill/Freeze \(Bill - Bill Segments\)](#)  
[Cancel \(Bill - Bill Segments\)](#)

### Generate (Bill - Bill Segments)

The **Generate** button is used to delete and recreate one or more bill segments. This button is enabled if you select at least one row from the bill segments grid that's in the **Error**, **Freezable** and/or **Incomplete** state.

Note that you can only select a bill segment for regeneration if ALL bill segments in the grid are in the **Error**, **Freezable** and/or **Incomplete** states. Note, you can use the **Status Filter** to restrict the type of bill segments in the grid.

You must specify the following parameters in order to regenerate a new bill:

- **Accounting Date** defines the financial period to which the new bill segments' financial transactions will be booked. The current date defaults when the window opens.
- Choose **Use Cut off Date** as **Billing Option**.

After specifying the parameters, click **Calculate** to regenerate the selected bill segments. A summary of what the system does is presented next.

### Freeze (Bill - Bill Segments)

The **Freeze** button is used to freeze one or more bill segments. This button is enabled if you select at least one row from the bill segments grid that's in the **Freezable** state AND the [installation option](#) indicates users are allowed to "freeze at will".

Note that you can only select a bill segment for freezing if ALL bill segments in the grid are in the **Freezable** state. Note, you can use the **Status Filter** to restrict the type of bill segments in the grid.

**Accounting Date** is used to define the financial period to which the frozen bill segments' financial transactions are booked. The current date defaults when the window opens.

After specifying the above parameters, click **Freeze** to freeze the selected bill segments. A summary of what the system does is presented next.

### Delete (Bill - Bill Segments)

The **Delete** button is used to delete one or more bill segments. This button is enabled if you select at least one row from the bill segments grid that's in the **Error**, **Freezable** and/or **Incomplete** state.

Note that you can only select a bill segment for deletion if ALL bill segments in the grid are in the **Error**, **Freezable** and/or **Incomplete** states. Note, you can use the **Status Filter** to restrict the type of bill segments in the grid.

If the system is successful in deleting the selected bill segment(s), a confirmation window summarizes what the system just did (i.e., it shows the number of bill segments that were deleted).

### Cancel/Rebill/Freeze (Bill - Bill Segments)

If problems are detected with a **frozen** bill segment, it should be canceled and a new bill segment should be created. We refer to this process as cancel / rebill. Refer to [Cancel / Rebill Incorrect Bill Segments](#) for more information.

Before you cancel / rebill, you'll probably need to fix the cause of the problem. The following table describes common problems and how to fix them.

Problem	How To Fix It
The wrong rate was used	Go to <a href="#">Obligation – Rate Info</a> and correct the rate.
The wrong tax exemption amount was calculated	Go to <a href="#">Obligation – Rate Info</a> and correct the tax exemption information.

**Multi-Cancel/Rebill Saves Time.** If the cause of the problem impacts many bill segments related to an obligation, you should use the [Multi Cancel Rebill](#) transaction as it allows you to cancel / rebill an unlimited number of bill segments at once.

The **Cancel/Rebill/Freeze** button is used to cancel, rebill and freeze one or more bill segments. This button is enabled if you select at least one row from the bill segments grid that's in the **Frozen** state AND the bill is not written off.

Note that you can only select a bill segment for cancel/rebill/freeze if ALL bill segments in the grid are in the **Frozen** state. Note, you can use the **Status Filter** to restrict the type of bill segments in the grid.

- **Cancel Reason** defines why the bill segment(s) are being canceled.
- **Accounting Date** defines the financial period to which the canceled and new bill segments' financial transactions are booked. The current date defaults when the window opens.
- Choose **Use Cut off Date** as **Billing Option**.
- If you prefer to have the system use the details used on the original bill segments, check **Use Old RQ**.

**Warning!** If the cause of the rebill is an incorrect rate and the new rate requires rate quantities that were not calculated when the bill was originally calculated, do not turn on **Use Old RQ**. Why? Because you want the system to derive new rate quantities during the rebill process.

After specifying the parameters, click **Calculate** to cancel, rebill and freeze the selected bill segments.

**Errors.** Bill segment errors may occur when you cancel / rebill the bill segments. For example, if you change the obligation's rate to a rate that is no longer effective on the bill segment period, a bill segment error is generated. If this occurs, you must go to [Bill Segment – Main](#) for the **pending cancel** bill segment and use the **Undo** action to remove the offending bill segment (and restore the original segment to the **Frozen** state). Alternatively, you can drill down on the **Error** bill segment and use the **Regenerate** action after correcting the cause of the problem.

**There is no Undo.** Unlike cancel / rebills performed on [Bill Segment – Main](#), there is no Undo action. This is because this transaction freezes the newly created bill segments after it cancel/rebills; whereas the bill segment transaction lets you examine the new bill segment before you freeze it. If you made a mistake, simply correct the cause of the mistake and then cancel / rebill again. The erroneous financial transactions are automatically suppressed on the next bill produced for the taxpayer.

A confirmation window summarizes what the system just did (i.e., it shows the number of bill segments that were deleted / generated).

## Cancel (Bill - Bill Segments)

The **Cancel** button is used to cancel one or more bill segments. This button is enabled if you select at least one row from the bill segments grid that's in the **Frozen** state AND the bill is not written off.

Note that you can only select a bill segment for cancellation if ALL bill segments in the grid are in the **Frozen** state. Note, you can use the **Status Filter** to restrict the type of bill segments in the grid.

- **Cancel Reason** defines why the bill segment(s) are being canceled.
- **Accounting Date** is used to define the financial period to which the canceled bill segments' financial transactions are booked. The current date defaults when the window opens.

After specifying the above parameters, click **Calculate** to cancel the selected bill segments.

**There is no Undo.** Unlike cancellations performed on [Bill Segment – Main](#), there is no Undo action. This transaction moves the pending cancel bill segments to the cancel state immediately; whereas the bill segment transaction lets you examine the cancellation before you commit it. If you cancelled bill segments by mistake, you must generate and freeze new bill segments.

## Bill - Bill Routings

This page is used to view the recipients of a bill. Refer to [Printing Bills](#) for a discussion of how this information is used to produce bill images.

This page is also used to request a new copy of a bill. Refer to [How To Reprint A Bill \(For The Original Recipients or For Someone New\)](#) for a description of how to do this.

**No routing information until completion.** A bill has no routing details until it is completed. At completion time, the system creates a routing detail for every person linked to the account that receives a copy of its bills. Refer to [The Source Of Bill Routing Information](#) for more information about how this information is constructed.

Use **Financial, Bill, Bill Routings** to open this page.

### Description of Page

**Bill Info** contains a concatenation of important information about the bill (e.g., the bill date, its status, due date, ending balance).

**Bill ID** is the system-assigned unique identifier of the bill.

The **Bill Routing Information** scroll contains an entry for each recipient of the bill. This information is disabled after the bill has been routed to the recipient. The remainder of this discussion describes the fields that can be defined for each recipient of the bill.

**Person ID** identifies the person who was originally associated with the routing record. You may change the **Person ID** to any person in the system as this information is only used to default the recipient's name and address information, below.

The **Bill Routing Parameters** area contains information explaining how and when the bill is routed to the recipient. This information is gray when the bill routing information has been extracted or if a bill is reopened:

- **Bill Route Type** is the method used to transmit the bill to the taxpayer. If the **Person ID** is linked to the account, the bill route type defaults from the person's [Account – Person Information](#).

- **Sequence** is the system-assigned identifier assigned to this bill routing.
- **Batch Control** is the identifier of the batch of bills in which the recipient's copy was downloaded. The batch process is defined on the **Bill Route Type**.
- **Extract Date/Time** is the date and time the bill was downloaded. This information only appears after the download has commenced.
- **Format** indicates if the taxpayer should receive a **Detailed** or a **Summary** bill. If the **Person ID** is linked to the account, this information defaults from [Account – Person Information](#).
- **Reprint** is checked if a user created this routing. This switch is populated by the system and is always protected.
- Turn on **Do Not Extract** if the bill should not be routed to the individual for whatever reason.
- **Copies** indicates the number of copies of the bill the person receives. If the **Person ID** is linked to the account, this information defaults from [Account – Person Information](#).
- **PO ID** is the purchase order Id the taxpayer wants printed on their copy of the bill. If the **Person ID** is linked to the account, this information defaults from [Account – Person Information](#).
- **Intercept** is the user ID of the individual who wants to review the bill before it is sent to the taxpayer. This field is only populated when the account has a Bill Print Intercept specified. Intercept information is defined on [Account – Main Information](#).

The **Mailing Address Information** area contains the recipient's name and address.

- If the recipient has an [override mailing name](#), **Name1**, **Name2** and **Name3** contain the person's override mailing name. Otherwise, **Name1** contains the recipient's primary name (this name might also include a prefix or suffix if the related [Account – Person](#) relationship has these fields defined).
- The remaining fields are the recipient's address. Refer to [The Source Of Bill Routing Information](#) for information about how the system constructs the recipient's address. Note:
  - If the recipient's address is a mailing address (as opposed to fax or email), the **Country** associated with the address controls the fields that appear in the address. Refer to [Defining Countries](#) for more information on the address constituents.
  - If a user manually adds a routing record, the mailing address information defaults from the information on the current routing record.
  - If a user specifies a **Person ID** that is linked to the bill's account, the **Bill Route Type** defined on the [Account – Person](#) record is defaulted.
  - If a user specifies a **Person ID** that is not linked to the bill's account, the **Bill Route Type** on the [installation record](#) is defaulted. Note, if the bill route indicates the bill is routed via the postal service, the system will only be able to default the person's address if the person has an override mailing address on their [correspondence information](#).
  - If the user changes the **Bill Route Type** after the initial default, the system populates the address information as described under [The Source Of Bill Routing Information](#).

## Bill - Bill Messages

The Messages page is a grid containing one row for every message that appears on the bill. Open **Financial, Bill, Bill Messages** to view this information.



**No messages until completion.** A bill has no messages until it is completed (unless you insert them manually). At completion time, the system assembles messages from the various message sources. Refer to [The Source Of Bill Messages](#) for information about these sources.

**The bill segment(s) may also have messages.** Be aware that only account-oriented messages are linked to a bill. There may also be obligation-oriented messages linked to the bill's bill segments. Refer to [Bill - Bill Messages](#) for information about the page on which obligation-oriented messages are displayed. Refer to [The Source Of Bill Messages](#) for information about the various message sources and whether each is linked to a bill or a bill segment.

**Bill messages may contain dynamic information.** Refer to [Substituting Field Values Into A Bill Message](#) for more information.

The Description of Page section that appears below simply describes the fields on this page. Refer to [How To Add Ad Hoc Messages To A Bill](#) for a description of how to perform common maintenance functions.

### Description of Page

**Bill Info** contains a concatenation of important information about the bill (e.g., the bill date, its status, due date, ending balance).

**Bill ID** is the system-assigned unique identifier of the bill.

The following columns are displayed in the grid:

- **Bill Message** is the code that identifies the bill message.
- **Message On Bill** is the message associated with the code.
- **Priority** is the bill messages priority (on the printed bill).
- **Insert Code** defines if the bill message causes an insert in the envelope.

## Bill - Characteristics

The Characteristics page is a grid containing one row for each characteristic linked to the bill. Bill characteristics are purely informational.

**Note.** You can only choose characteristic types defined as permissible on a bill record. Refer to [Setting Up Characteristic Types and Their Values](#) for more information.

### Description of Page

The following fields display on the grid:

- **Characteristic Type** must be a type of characteristic defined as permissible on a bill record.
- **Characteristic Value** indicates the value of the characteristic if the characteristic type is pre-defined or foreign key reference.



## How To

The topics in this section describe how to perform common bill maintenance functions.

### Contents

- [How To Create A Bill For All Obligations Linked To An Account](#)
- [How To Create A Bill For A Specific Obligation](#)
- [How To Create a Bill with no Bill Segments](#)
- [How To Create An Ad-hoc Bill](#)
- [How To Correct A Bill Segment That's In Error](#)
- [How To Correct A Bill That's In Error](#)
- [How To Cancel A Bill Segment](#)
- [How To Cancel / Rebill A Bill Segment](#)
- [How To Remove Unwanted Adjustments \(or Payments\) From A Completed Bill](#)
- [How To Add An Adjustment \(or Payment\) To A Completed Bill](#)
- [How To Reprint A Bill \(For The Original Recipients or For Someone New\)](#)
- [How To Add Ad Hoc Messages To A Bill](#)
- [How To Remove A Completed Bill](#)
- [How To Display A Bill On-line](#)

### How To Create A Bill For All Obligations Linked To An Account

99.9% of all bills are created by the batch bill process and require no human intervention before they are finalized and sent to a taxpayer. The other 0.1% are created by users on-line / real time. This section explains how to create the 0.1%.

The following points describe how to use this page to create a new bill:

- Navigate to the [Bill – Main Information](#) page:
  - If you have already selected the taxpayer (say on Control Central), select **Go To Bill +** option on the account [context menu](#). This will transfer you to this page with the respective account populated.
  - If you haven't selected the account, use the main menu to navigate to this page (Main Menu, Financial, Bill, +) and then select the **Account ID** using the adjacent [search button](#).
- Click the [Generate](#) button to create a new bill.
- You will see a summary of the bill segments in the lower right portion of the page. If there are errors, refer to [How To Correct A Bill Segment That's In Error](#). If there are no errors, the **Bill Segment Freeze Option** on the [installation option](#) controls the next step:
  - If this option is set to **Freeze At Bill Completion**, click the [Freeze / Complete](#) button.
  - If this option is set to **Freeze At Will**, click the [Freeze](#) button and, if everything looks clean, click the [Complete](#) button.

### How To Create A Bill For A Specific Obligation

Most bills contain a bill segment for every billable obligation linked to the account. If you want to produce a bill that contains bill segments for a subset of obligations (e.g., you want to create an ad-hoc bill for an account's billable charge obligation), follow these steps:

- Navigate to the [Bill – Main Information](#) page:

- If you have already selected the taxpayer (say on Control Central), select **Go To Bill +** option on the account [context menu](#). This will transfer you to this page with the respective account populated.
- If you haven't selected the account, use the main menu to navigate to this page (Main Menu, Financial, Bill, +) and then select the **Account ID** using the adjacent [search button](#).
- Click the [save](#) button to create a bill without bill segments for the account.
- Select **Go To Bill Segment +** option on the Bill's [context menu](#) to transfer to [Bill Segment - Main](#).
- Select the **Obligation ID** of the obligation for which you want to create a bill segment.
- Click the [Generate](#) button to generate a bill segment.
- If an error exists, refer to [How To Correct A Bill Segment That's In Error](#).
- If there are no errors, the **Bill Segment Freeze Option** on the [installation option](#) controls the next step:
  - If this option is set to **Freeze At Bill Completion**, return to [Bill – Main](#) (using the bill context menu) and click the [Freeze / Complete](#) button.
  - If this option is set to **Freeze At Will**, click the [Freeze](#) button and, if everything looks clean, return to [Bill – Main](#) and click the [Complete](#) button.

## How To Create a Bill with no Bill Segments

You may wish to generate a bill that contains no bill segments. For example, perhaps your taxpayer is billed quarterly but pays monthly. During the months where no charges are calculated, you may choose to send a "bill" that does not show new charges, but includes the payments received. To produce a bill with no bill segments, follow these steps:

- Navigate to the [Bill – Main Information](#) page:
  - If you have already selected the taxpayer (say on Control Central), select **Go To Bill +** option on the account [context menu](#). This will transfer you to this page with the respective account populated.
  - If you haven't selected the account, use the main menu to navigate to this page (Main Menu, Financial, Bill, +) and then select the **Account ID** using the adjacent [search button](#).
- Click the [save](#) button to create a bill without bill segments for the account.
- At this point either the **Complete** or **Freeze / Complete** button (based on the value of Bill Segment Freeze Option on [installation options](#)) becomes enabled. Click the applicable button for your installation to complete the bill. At this point, all the steps described for completing a bill in Bill Lifecycle are performed, including sweeping onto the bill any payments received since the last bill was generated.

## How To Create An Ad-hoc Bill

The following points describe the steps necessary to create and bill an ad-hoc bill:

- If the taxpayer to be invoiced doesn't already have an appropriate billable charge obligation, create one.

- Use [Maintaining Billable Charges](#) to add a billable charge for the obligation. The billable charge contains the invoice lines and amounts.
- If you created a new billable charge obligation in step 1, you must activate it before billing can bill it. To do this, transfer to [Obligation – Main Information](#) and click the activate button.

Refer to [How To Create A Bill For A Specific Obligation](#) for instructions describing how to create a bill with a bill segment for the billable charge obligation.

## How To Correct A Bill Segment That's In Error

A bill segment will exist in the **Error** state if a problem was encountered during the generation of a bill segment. To fix an **error** bill segment you must correct the cause of the error and then regenerate the bill segment. Refer to [Bill Segment Errors](#) for background information.

**Error segments created during batch billing.** If the **error** bill segment was created as part of the cyclical batch bill process, the system attempts to fix the offending segment(s) each night during the account's bill window by regenerating it. Therefore, if the cause of the error is fixed during the day, the system will automatically regenerate the bill segment when batch billing next runs; you don't have to manually correct each bill. And once a bill is error-free, it will be completed and sent to the taxpayer. Refer to [Batch Billing](#) for more information.

Correcting the cause of a bill segment error can be difficult or trivial; we hope that the error messages (along with their long descriptions) provide a good description of how to resolve the issue.

After you have corrected the cause of the error, you should regenerate the bill segment. There are two ways to do this:

- You can use the [Generate](#) action on [Bill – Bill Segment](#) to regenerate one or more bill segments. We recommend using this method if the cause of the error affected many bill segments because you can regenerate many bill segments at a time on this page. Also, be aware that this transaction shows the estimated value of **error** bill segments so you can gauge whether to invest a lot of time in correcting the problem. For example, if the bill segment's estimated value is \$10,000, it's probably worth correcting the problem before sending the bill out. However, if the bill segment's value is small, you may decide to simply delete the error bill segment and send the bill without it (the system will attempt to create a bill for the deleted period the next time the account is billed).
- You can use the [Generate](#) action on [Bill Segment - Main](#) to regenerate an individual bill segment. We'd recommend using this method if the problem is bill segment-specific and you need to examine the information that is snapshot on the bill segment to resolve the error.

**Drilling Into Bill Segment Errors.** When you use the bill segments in error To Do list to work through errors, you will be transferred to [Bill Segment - Main](#) with the bill segment displayed.

After all errors have been resolved on a bill, return to [Bill – Main](#). The **Bill Segment Freeze Option** on the [installation option](#) controls the next step:

- If this option is set to **Freeze At Bill Completion**, click the [Freeze / Complete](#) button.
- If this option is set to **Freeze At Will**, click the [Freeze](#) button and, if everything looks clean, click the [Complete](#) button.

## How To Correct A Bill That's In Error

Besides errors on bill segments, there may also be errors detected when the system attempts to complete a bill. For example, if the system cannot find a mailing address, the bill will be in error (as opposed to one of its bill segments). Refer to [Bill Errors](#) for background information.

To correct such an error:

- Navigate to [Bill – Main](#).
- Look at the bill's error message. Push the view button to view the long explanation that will provide suggestions as to the cause of the error (and how to fix it). Transfer to the appropriate page to fix the problem.
- After fixing the cause of the error, return to [Bill – Main](#). The **Bill Segment Freeze Option** on the [installation option](#) controls the next step:
  - If this option is set to **Freeze At Bill Completion**, click the [Freeze / Complete](#) button.
  - If this option is set to **Freeze At Will**, click the [Freeze](#) button and, if everything looks clean, click the [Complete](#) button.

## How To Cancel A Bill Segment

If a bill segment was **frozen** and it should never have been created in the first place, the bill segment must be canceled. There are several ways to do this:

- You can use the [Cancel Frozen](#) action on [Bill – Main](#) to cancel all frozen bill segments linked to the bill. We recommend using this method if you need to cancel everything on a bill.
- You can use the [Cancel](#) action on [Bill – Bill Segments](#) to cancel selected bill segments. We recommend using this method if you need to cancel multiple bill segments.
- You can use the [Cancel](#) action on [Bill Segment - Main](#) to cancel an individual bill segment. We'd recommend using this method if you need to examine the information on the bill segment before canceling it.

If the related bill is **pending**, the cancellation will cause the bill segment to be suppressed on the version of the bill sent to the taxpayer (but it remains in the database for audit and financial reporting purposes).

If the related bill is **complete** and you do not [Reopen](#) the bill, the financial impact of the canceled bill segment(s) will appear on the next bill created for the account (under Bill Corrections).

If the related bill is **complete** you may be able to [Reopen](#) the bill and then [Complete](#) it. By doing this, you can suppress a **frozen** bill segment on a previously completed bill. This means that if you catch a problem after completion you don't necessarily have to show the problem to the taxpayer. Rather, cancel the problem, reopen and then recomplete the bill (when you recomplete the bill the system will mark the bill segment to be suppressed on the version sent to the taxpayer because its cancellation appears on the same bill as the original charges).

## How To Cancel / Rebill A Bill Segment

If a bill segment was **frozen** but it contains incorrect information, it should be canceled and a new bill segment should be created. We refer to this process as cancel / rebill. Refer to [Cancel / Rebill Incorrect Bill Segments](#) for more information.

Before you cancel / rebill, you must correct the cause of the problem. The following table describes common problems and where to go to fix them.

Problem	How To Fix It
The wrong rate was used	Go to <a href="#">Obligation – Rate Info</a> and correct the rate.
The wrong tax exemption amount was calculated	Go to <a href="#">Obligation – Rate Info</a> and correct the tax exemption information.

After correcting the cause of the problem, you're ready to cancel and rebill the offending bill segment(s). There are several ways to do this:

- You can use the [Cancel / Rebill / Freeze](#) action on [Bill – Bill Segments](#) to cancel, rebill and freeze selected bill segments. We recommend using this method if you need to cancel / rebill multiple bill segments due to a pervasive problem.
- You can use the [Rebill](#) action on [Bill Segment - Main](#) to cancel and rebill an individual bill segment. We'd recommend using this method if you need to examine the information on the bill segment before and after canceling it. If you use this method, you must also use the [Freeze](#) action to freeze the newly created bill segment.
- If the cause of the problem impacts many bill segments related to an obligation, you should use the [Multi Cancel Rebill](#) transaction as it allows you to cancel / rebill an unlimited number of bill segments at once.

If the related bill is **pending**, the cancellation will cause the erroneous bill segment to be suppressed on the version of the bill sent to the taxpayer (but it remains in the database for audit and financial reporting purposes).

If the related bill is **complete** and you do not [Reopen](#) the bill, the financial impact of the canceled bill segment(s) and their rebill bill segment(s) will appear on the next bill created for the account (under Bill Corrections).

If the related bill is **complete** you may be able to [Reopen](#) the bill and then [Complete](#) it. By doing this, you can suppress an erroneous **frozen** bill segment on a previously completed bill. This means that if you catch a problem after completion you don't necessarily have to show the problem to the taxpayer. Rather, cancel / rebill the problem, reopen and then recomplete the bill (when you recomplete the bill the system will mark the bill segment to be suppressed on the version sent to the taxpayer because its cancellation appears on the same bill as the original charges).

## How To Remove Unwanted Adjustments (or Payments) From A Completed Bill

If the system (or a user) has created adjustments that have been swept onto a recently completed bill but you don't want them to appear on the bill perform the following steps:

- Cancel the adjustment ([Adjustments – Main Information](#)).
- Navigate to the [Bill - Main](#) page.
- Click the [Reopen](#) button to reopen the previously completed bill.
- Click the [Complete](#) button to recomplete the bill. When the bill is recompleted, the system will see that an adjustment that was swept onto the bill when it was originally completed has been canceled and it will mark the adjustment for suppression on the printed bill.

You can remove unwanted payments using an analogous approach – Cancel the payment, Reopen the bill, Complete the bill.

**Cannot reopen historical bills.** You may only reopen an account's most recent bill because recompleting the bill causes the ending-balance to change, and we don't want this to happen to historical bills.

## How To Add An Adjustment (or Payment) To A Completed Bill

If you want to add an adjustment to a completed bill perform the following steps:

- Add and freeze the adjustment ([Adjustments – Main Information](#)).
- Navigate to the [Bill - Main](#) page.
- Click the [Reopen](#) button to reopen the previously completed bill.
- Click the [Complete](#) button to recomplete the bill. When the bill is recompleted, the system will sweep the recently created financial transactions onto itself.

You can add a payment to a bill using an analogous approach – Add the payment, Reopen the bill, Complete the bill.

**Cannot reopen historical bills.** You may only reopen an account's most recent bill because recompleting the bill causes the ending-balance to change, and we don't want this to happen to historical bills.

## How To Reprint A Bill (For The Original Recipients or For Someone New)

A copy of a bill is sent to every individual / business specified in the [bill's routing details](#). The system creates the routing details when a bill is completed. Refer to [The Source Of Bill Routing Information](#) for more information about how this information is compiled during bill completion.

If you want to reprint a bill OR if you want to send a bill to someone other than the original recipient(s):

- Navigate to the [Bill - Main](#) page.
- Transfer to the [Bill - Bill Routings](#) tab.
- Insert a new row (click the + button). Note well, you can only insert information on a bill that is **Complete**.
- Specify the name and address to which the bill should be sent. If you want to send the bill to someone other than the original recipient, simply specify the individual's name and address. If the recipient is a person in the system, you can select their person ID and let the system default this information for you.
- Save the bill.

If a bill is **Complete** but the print process hasn't yet executed, you don't have to insert a new routing row. Rather, you can simply change the name and address on the original routing details.

## How To Add Ad Hoc Messages To A Bill

If you want to add a message to a completed bill:

- Navigate to the [Bill - Main](#) page.
- Transfer to the [Bill - Bill Messages](#) tab.
- Insert a new row (click the + button). Note, you can only insert a message on the most recent bill that is **Complete**.
- Specify the desired message code.
- Save the bill.
- Reprint the bill as per [How To Reprint A Bill \(For The Original Recipients or For Someone New\)](#).

Refer to [The Source Of Bill Messages](#) for more information about the messages the system automatically links to a bill when the bill is completed.

### How To Remove A Completed Bill

You can't physically remove a **completed** bill from the system. However, you can remove all financial transactions from the bill (and therefore end up with a zero value bill). You can also suppress the bill from being printed. To do this, perform the following steps:

- If the bill has bill segments, cancel each bill segment. Refer to [How To Cancel A Bill Segment](#) for instructions describing how to do this.
- If the bill has payment amounts or adjustment amounts that were entered incorrectly, navigate to the payment and/or adjustment pages and cancel the payments / adjustments.
- Navigate to the [Bill - Main](#) page.
- Click the [Reopen](#) button to reopen the previously completed bill.
- Transfer to the [Bill - Bill Routings](#) tab. Turn on the Do Not Print switch for every routing entry.
- Click the [Complete](#) button to recomplete the bill. When the bill is recompleted, the system will sweep the canceled segments onto it and the net effect should be zero current charges.

**Cannot reopen historical bills.** You may only reopen an account's most recent bill because recompleting the bill causes the ending balance to change, and we don't want this to happen to historical bills.

If the bill has payment amounts or adjustment amounts that are valid, but you don't want them to appear on this bill, navigate to the **Bill, Financial Details** page. Drill over to each respective financial transaction by clicking the drill down button. Clicking this button transfers you to [Financial Transaction - Main](#). On this page, clear out the value of the Bill ID. Clearing out the Bill ID unlinks the financial transaction from the bill. When the system next completes a bill for the account, it will find the unbilled financial transaction and link it to the bill.

### How To Display A Bill On-line

To display a bill on-line, you must first have installed software to render the bill in a format suitable for displaying. You must also configure the system to invoke this software; otherwise, you will get a message indicating that the service is not available. This option can only be configured by your technical staff.



To view a bill on-line, click the **Display Bill** button.

## Financial Transactions On A Bill

Open **Financial Query, Financial Transactions On A Bill** to view the financial transactions on a bill.

**Note.** You can also open this page using the go to buttons that prefix the financial transaction summaries on [Bill - Main](#).

### Description of page

**Bill Id** is the system-assigned unique identifier of the bill.

**Account ID** is the bill's account.

The area beneath **Account ID** provides you with options that control which financial transactions appear in the grid. The following points describe the various options:

- Use the **Obligation Filter** to define the types of obligations whose financial transactions appear in the grid. The following options are available:
  - **All.** Use this option if you do not wish to restrict financial transactions based on obligation attributes.
  - **Obligation ID.** Use this option to restrict financial transactions to those of a specific obligation.
  - **Obligation Type.** Use this option to restrict financial transactions to those whose obligations are linked to a given **division** and **obligation type**.
- Use **FT Type Filter** to restrict the type of financial transactions that appear in the grid. The following options are available:
  - **Adjustments.** This option shows all financial transactions associated with adjustments.
  - **All.** This option shows all financial transactions.
  - **Corrections.** This option shows all financial transactions associated with corrections (i.e., cancel / rebills that have occurred after the bill was completed).
  - **Current Charges.** This option shows all financial transactions associated with bill segments (that aren't **Corrections**).
  - **Payments.** This option shows all financial transactions associated with payments.

Don't forget to click the search button after changing the filters or after selecting a new Bill ID.

The grid that follows contains the financial transactions that match your search criteria. The following information is displayed:

- **FT Type** displays the type of financial transaction except for adjustments. For adjustments, the adjustment type's description is shown. Click on the hyperlink to transfer to [Financial Transaction - Main](#). On this page, you can change certain aspects of the FT in question. Refer to [How To Remove Unwanted Adjustments \(or Payments\) From A Completed Bill](#) for a description of how to perform common maintenance functions.



- **Accounting Date** is the date the system uses to determine the financial transaction's accounting period in your general ledger.
- **Current Amount** contains the financial transaction's effect on the obligation's current balance.
- **Payoff Amount** contains the financial transaction's effect on the obligation's payoff balance. The **Payoff Amount** will be dim if it equals the **Current Amount**.
- **Show on Bill** indicates if information about the financial transaction appears on the taxpayer's bill.
- **Obligation Information** contains a summary of the respective obligation.
- **Financial Transaction ID** is the system-assigned unique identifier of the financial transaction.

At the bottom of the page is a summary of the financial transactions that match the search criteria.

## Maintaining Bill Segments

A bill typically contains one bill segment for every active billable obligation linked to its account. A bill segment contains information showing how the segment was calculated and how it should be printed on the taxpayer's bill.

The actions taken to create a bill segment are dependent on the obligation's obligation type.

It's important to be aware that there are very few fields that are directly modifiable by a user. To modify most fields on a bill segment, you have to change source information (e.g., obligation, rate) and then regenerate the bill segment. For example, if you want to change the rate used to calculate a bill segment, you must change the rate history on the respective obligation and then cancel / rebill the bill segment. Refer to [How To](#) for step-by-step instructions that explain how to perform common bill segment maintenance functions.

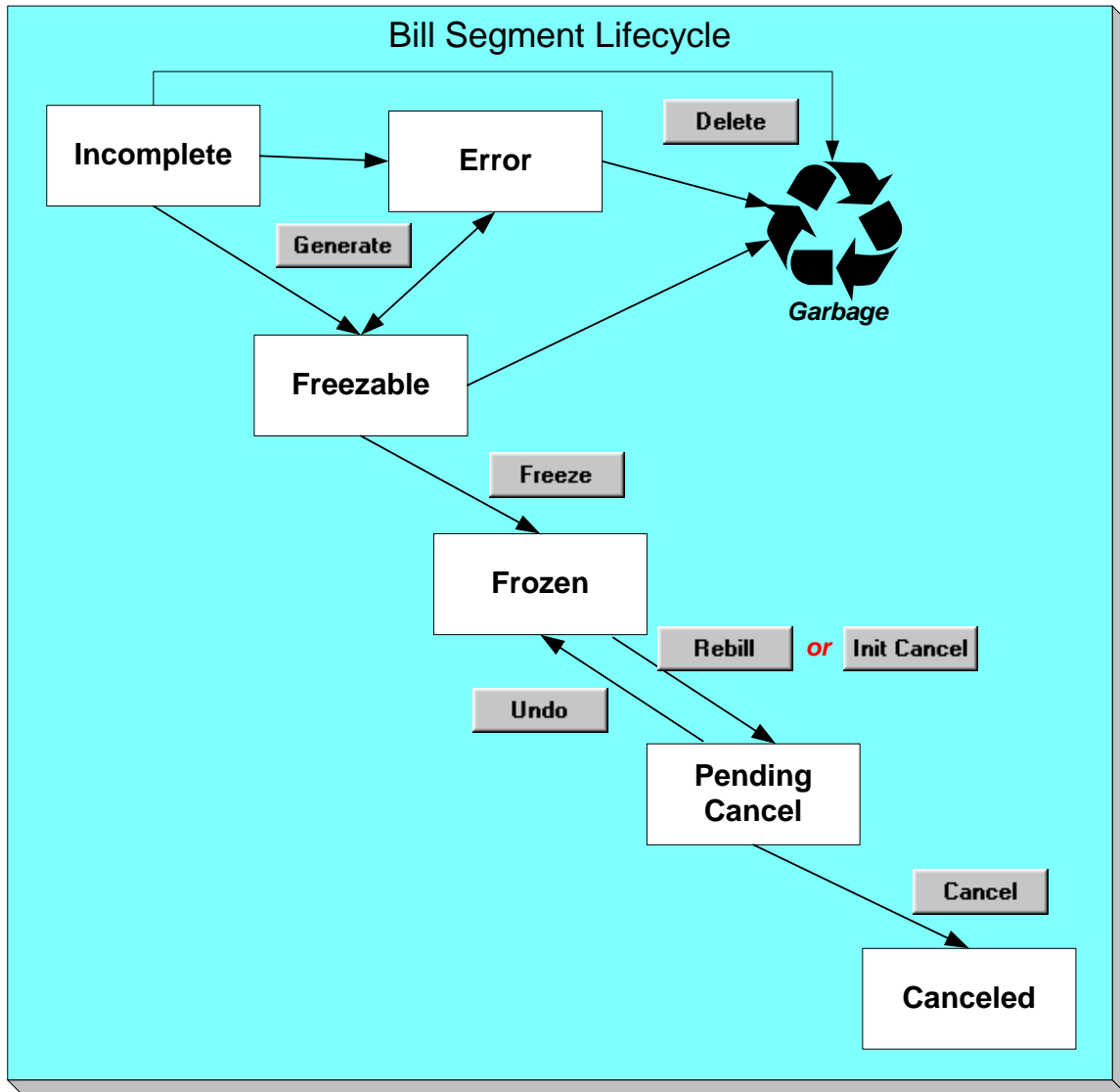
### Contents

- [Bill Segment Lifecycle](#)
- [Bill Segment - Main Information](#)
- [Bill Segment - RQ Details](#)
- [Bill Segment - Calc Lines](#)
- [Bill Segment - Financial Details](#)
- [Bill Segment - Bill Segment Messages](#)

## Bill Segment Lifecycle

The following diagram shows the possible lifecycle of a bill segment.

**Warning!** This diagram only makes sense in the context of the page used to maintain bill segments. Refer to [Bill Segment – Main Information](#) for the details.



Bill segments are initially created in the **Incomplete** state. Bill segments in this state don't have bill lines or a financial transaction; they are simply a stub awaiting generation.

Click **Generate** to generate the bill segment's bill lines and its financial transaction.

- If the system cannot generate the bill segment (for any number of reasons), the bill segment is moved to the **Error** state. You may regenerate the bill segment after correcting the source of the error (by clicking **Generate**). You may also delete such a bill segment.
- If the system successfully generates a bill segment, the bill segment becomes **Freezable**. If the bill segment is incorrect, correct the cause of the problem (e.g., fix the meter read, change the obligation's rate) and regenerate the bill segment (by clicking **Generate**).

Click the delete button to physically remove an **Incomplete**, **Error** or **Freezable** bill segment from the database.

Click **Freeze** to freeze a bill segment and its financial transaction. After doing this, the bill segment's state becomes **Frozen** and the bill segment may now appear on a taxpayer's bill. Also note, when a bill segment is frozen, its financial details will be interfaced to the general ledger the next time the [GL interface](#) executes.

You may not change a bill segment once it is frozen. However, you have two options if you want to change a frozen bill segment's financial impact:

- If you need to remove the financial effects of a bill segment, click **Init Cancel**. Clicking this button causes a new financial transaction to be generated and linked to the bill segment. This financial transaction reverses the financial effects of the original bill segment. The bill segment becomes **Pending Cancel** after the system successfully generates this financial transaction. At this point, you have two options:
  - Click **Cancel** to freeze the cancellation. Clicking this button causes the bill segment to move to the **Canceled** state. Once canceled, the bill segment cannot be uncanceled or changed.
  - Click **Undo** to return the pending cancellation to the **Frozen** state.
- Click **Rebill** if you need to recalculate the bill segment because something has changed since it was originally created (e.g., the obligation's rate was changed, a new read was entered). Clicking this button causes the following events to take place:
  - The original bill segment is moved to the **Pending Cancel** state.
  - A new bill segment is created in the **Freezable** (or **Error**) state.
 At this point, you have two options.
  - Click **Freeze** to make the new bill segment **Frozen** and make the original bill segment **Canceled**.

**Multi-Cancel/Rebill Saves Time.** If you need to rebill many bill segments related to an obligation (e.g., because many bill segments were created using the wrong rate), you should use the [Multi Cancel Rebill](#) transaction as it allows you to cancel / rebill an unlimited number of bill segments at once.

Click **Undo** to delete the new bill segment and return the original bill segment to the **Frozen** state.

## Bill Segment - Main Information

The Description of Page section that appears below simply describes the fields on this page. Refer to [How To](#) for a description of how to perform common bill maintenance functions.

The Main page contains core bill segment information. Open **Financial, Bill Segment, Main** to view this information.

**Note.** Most bill segments are created by a background process. Refer to [A High Level Overview Of The Bill Creation Process](#) for information describing how the system creates a bill segment.

### Description of Page

**Bill Seg Info** is a concatenation of the bill segment's division, obligation type, status, bill period and amount. **Bill Segment ID** is the system-assigned unique identifier of the bill segment.

**Account ID** references the bill segment's account.

**Current Amount** is the bill segment's effect on the obligation's current balance. **Payoff Amount** is only shown if it differs from current amount. Refer to [Billing – Current Balance versus Payoff Balance](#) for more information.

**Bill ID** is the system-assigned unique identifier of the bill on which the bill segment appears. A concatenation of its bill date, status, due date and amount is displayed.

**Obligation ID** contains information about the bill segment's obligation. A concatenation of the obligation's division, obligation type, start option (if any), status and start date is displayed.

**Period** is the start and end dates of the bill segment.

**Bill Cycle** displays the bill cycle and window start date of the bill segment's bill. This information is only populated on bills when they are generated on-schedule by the batch billing background process. Bills generated manually, i.e. off-schedule, do not have this information.

**Status** is the bill segment's status. Refer to [Bill Segment Lifecycle](#) for the potential values and how to handle a bill segment when it exists in a given state.

The **Closing** switch is not used.

If the **RQ Override** switch is on, a user has overridden the rate quantities.

**Location** is the address of the main location associated with the obligation (as defined on the obligation's characteristic location).

**Create Dt/Tm** is the date and time on which the bill segment was created or regenerated.

In the **Message** area, a brief error message appears if there's a problem with the bill segment. The message area is suppressed if there are no problems with the bill.

Push the magnifying glass button to view the long explanation. The long explanation will provide information about the cause of the error (and how to fix it).

If the bill segment is a rebill of a canceled bill segment, a reference to the original bill segment is displayed. Click the adjacent go to button to transfer to the bill segment that was canceled.

If the bill segment was canceled and rebilled by another segment, a reference to the new bill segment is displayed. Click the adjacent go to button to transfer to the new bill segment that superseded the canceled segment.

The topics that follow describe each of the actions on that appear in the **Bill Segment Actions** area. Refer to [How To](#) for a description of typical business processes that use these buttons.

**Mass update actions.** This page allows you to work on individual bill segments. If you want to update many bill segments linked to a bill at once, try using the mass update actions available on [Bill -Bill Segments](#).

## Contents

- [Generate \(Bill Segment\)](#)
- [Delete \(Bill Segment\)](#)
- [Freeze \(Bill Segment\)](#)
- [Rebill \(Bill Segment\)](#)
- [Init Cancel \(Bill Segment\)](#)
- [Undo \(Bill Segment\)](#)
- [Cancel \(Bill Segment\)](#)

## Generate (Bill Segment)

Click the **Generate** button to create a new a bill segment.

This button is enabled when either of the following conditions is true:

- You are in add mode (i.e., you are not displaying an existing bill segment) and you have selected an **Account ID**, **Bill ID** and **Obligation ID**.
- You are displaying a bill segment that is **Incomplete**, **Error**, or **Freezable**. Refer to [Bill Segment Lifecycle](#) for more information about these status values.

You must specify the following parameters in order to generate a new bill segment:

- **Cut-off Date** is the last possible day of the bill segment's bill period. The current date defaults when the window opens. Refer to [Ways To Control The End Date Of A Bill Segment](#) for more information.
- **Accounting Date** is used to define the financial period to which the new bill segment's financial transaction will be booked. The current date defaults when the window opens.

After specifying the above parameters, click the **Calculate** button to create a new bill segment for the bill.

If an error exists, refer to [How To Correct A Bill Segment That's In Error](#).

If there are no errors, the **Bill Segment Freeze Option** on the [installation option](#) controls the next step:

- If this option is set to **Freeze At Bill Completion**, return to [Bill – Main](#) (using the bill context menu) and click the [Freeze / Complete](#) button.
- If this option is set to **Freeze At Will**, click the [Freeze](#) button and, if everything looks clean, return to [Bill – Main](#) and click the [Complete](#) button.

## Delete (Bill Segment)

The **Delete** button deletes a bill segment.

This button is enabled if you are displaying a bill segment that is **Incomplete**, **Error**, or **Freezable**. Refer to [Bill Segment Lifecycle](#) for more information about these status values.

## Freeze (Bill Segment)

Clicking the **Freeze** button causes a **freezable** bill segment to become **frozen**. Refer to [Bill Segment Lifecycle](#) for more information about freezing.

This button is enabled if:

- the **Freeze At Will Bill Segment Freeze Option** on the [installation option](#) has been selected AND
- the bill segment is **freezable**.

**Freezing a day or more after generation.** If the bill segment is linked to a **closed** accounting period, a pop-up appears in which you can specify a new accounting date. This will only happen if the accounting period closes before you freeze the bill segment (the accounting date is stamped on a bill segment when it's initially generated).

**If problems are detected after freezing.** A bill segment may not be changed after it is **frozen**. All subsequent changes must occur by canceling the frozen bill segment and creating a new bill segment. Refer to [How To Cancel A Bill Segment](#) and [How To Cancel / Rebill A Bill Segment](#) for more information.

## Rebill (Bill Segment)

If problems are detected with a **frozen** bill segment, it should be canceled and a new bill segment should be created. We refer to this process as cancel / rebill. Refer to [Cancel / Rebill Incorrect Bill Segments](#) for more information.

Before you cancel / rebill, you'll probably need to fix the cause of the problem. The following table describes common problems and where to go to fix them.

Problem	How To Fix It
The wrong rate was used	Go to <a href="#">Obligation – Rate Info</a> and correct the rate.
The wrong tax exemption amount was calculated	Go to <a href="#">Obligation – Rate Info</a> and correct the tax exemption information.

**Multi-Cancel/Rebill Saves Time.** If the cause of the problem impacts many bill segments related to an obligation, you should use the [Multi Cancel Rebill](#) transaction as it allows you to cancel / rebill an unlimited number of bill segments at once.

The **Rebill** button causes an existing bill segment to be canceled, and a new bill segment to be created.

This button is enabled when you display a **Frozen** bill segment AND the bill segment's bill is not written off. Refer to [Bill Segment Lifecycle](#) for more information about these status values.

You must specify the following parameters in order to cancel and rebill a new bill segment:

- **Cancel Reason** defines why the bill segment is being canceled.
- **Cutoff Date** is the last day of the new bill segment. The cutoff date on the bill segment defaults when the window opens.
- **Accounting Date** is used to define the financial period to which the canceled and new bill segments' financial transactions will be booked. The current date defaults when the window opens.
- Choose **Use Cut off Date** as **Billing Option**.
- Turn on **Use Old RQ for Regen** if you want the system to use the details on the original bill segment when it calculates the new bill segment.

Click the **Calculate** button to cancel the original bill segment and create a new bill segment.

At this point, there are now two bill segments - the original bill segment is in the state of **Pending Cancel**, the new bill segment is in the **Freezable** (or **Error**) state. If you want to back out, click the go to button to return to the original bill segment and then click [Undo](#) (this will delete the new bill segment and return the original bill segment to the **Frozen** state). If the new bill segment looks correct, click [Freeze](#) to move the original bill segment to the **Canceled** state and the new bill segment to the **Frozen** state.

### Init Cancel (Bill Segment)

The **Init Cancel** button causes the first step of the bill segment cancellation process to be executed. You'd click this button if a **frozen** bill segment should never have been created (i.e., you want to remove the financial impact of a bill segment from a taxpayer's balance).

This button is enabled when you display a **Frozen** bill segment AND the bill segment's bill is not written off. Refer to [Bill Segment Lifecycle](#) for more information about these status values.

You must specify the following parameters in order to cancel and rebill a new bill segment:

- **Cancel Reason** defines why the bill segment is being canceled.
- **Accounting Date** is used to define the financial period to which the canceled bill segment's financial transactions will be booked. The current date defaults when the window opens.

Click the **Calculate** button to cancel the original bill segment.

At this point, the bill segment is in the state of **Pending Cancel**. If you want to back out, click [Undo](#); the bill segment will be returned to the **Frozen** state. Click [Cancel](#) to move the bill segment to the **Canceled** state.

### Undo (Bill Segment)

The **Undo** button returns a **Pending Cancel** bill segment to the **Frozen** state. If a **Pending Cancel** bill segment was created during a Cancel / Rebill, the newly created **Freezable** bill segment is also deleted.

This button is enabled when you display a **Pending Cancel** bill segment. Refer to [Bill Segment Lifecycle](#) for more information about this status.

### Cancel (Bill Segment)

Clicking the **Cancel** button causes a **Pending Cancel** bill segment to become **Canceled**. Refer to [Bill Segment Lifecycle](#) for more information about cancellation.

This button is enabled if a **Pending Cancel** bill segment is displayed AND the bill segment's bill is not written off.

You must specify the following parameters in order to cancel the frozen bill segments:

- **Cancel Reason** defines why you are performing the cancellation.
- **Accounting Date** defines the financial period to which the canceled bill segments' financial transactions will be booked. The current date defaults when the window opens.

After specifying the above parameters, click the **Calculate** button to cancel the frozen bill segments.

If the related bill is **pending**, the cancellation will cause the bill segment to be suppressed on the version of the bill sent to the taxpayer (but it remains in the database for audit and financial reporting purposes).

If the related bill is **complete** and you do not [Reopen](#) the bill, the financial impact of the canceled bill segment will appear on the next bill created for the account (under Bill Corrections).

If the related bill is **complete** you may be able to [Reopen](#) the bill and then [Complete](#) it. By doing this, you can suppress a **frozen** bill segment on a previously completed bill. This means that if you catch a problem after completion you don't necessarily have to show the problem to the taxpayer. Rather, cancel the problem, reopen and then recomplete the bill (when you recomplete the bill the system will mark the bill segment to be suppressed on the version sent to the taxpayer because its cancellation appears on the same bill as the original charges).

## Bill Segment - RQ Details

The Description of Page section that appears below simply describes the fields on this page.

The RQ Details page contains information about the rate quantities (RQ) that will be priced by the obligation's rate. In addition to showing RQ details, this page shows any Calculation / Audit Read Details that are linked to the bill segment. Open **Financial, Bill Segment, RQ Details** to view this information.

### Description of Page

**Bill Seg Info** is a concatenation of the bill segment's division, obligation type, status, bill period and amount. **Bill Segment ID** is the system-assigned unique identifier of the bill segment.

The **Rate Quantity** details grid is a snapshot of the rate quantities amassed from:

- The RQ rules defined on the obligation's rate. Refer to [Defining Measured Quantity Manipulation Rules](#) for more information about how RQ rules can introduce additional rate quantities.
- [Billable charges](#) that supply rate quantities.

One row exists for every unique combination of unit of measure (UOM), time-of-use (TOU) code, and rate quantity identifier (RQI) associated with the obligation.

This information may be overridden on a bill segment in the **Error** or **Freezable** state. Insert one row in the rate quantity grid (click the + button) for each UOM/TOU/RQI you need to add. You may also change amounts that were populated by the system when it initially generated the bill by simply overwriting the information.

**Warning!** The proper way to fix the RQ details on a bill segment is to correct the cause of the error. Overriding the RQ details on a bill segment is a last resort that should only be used when you can't fix the cause of the problem. Refer to [How To Cancel / Rebill A Bill Segment](#) for ways to fix common problems.

The following information is displayed in the grid:

- **Unit of Measure** is the unit of measure of the rate quantity.
- **Time of Use** is the time-of-use of the rate quantity.
- **RQI** is the rate quantity identifier of the rate quantity.
- **Initial Rate Quantity** is the initial quantity amassed by the system before application of the rate's RQ rules.
- **Billable Rate Quantity** is the quantity that will be priced by the obligation's rate. This amount differs from the initial quantity when the rate's RQ rules manipulate the value.



The **Calculation/Audit Read Details** section contains Calculation / Audit Read Details. This section only appears if the obligation has a special role of **Billable Charge**.

The following information may be defined for each calculation / audit read:

- **Final Unit of Measure** is the final unit of measure of the calculation / audit read.
- **Final Time of Use** is the final time-of-use code of the calculation / audit read.
- **Final RQI** is the final rate quantity identifier of the calculation / audit read.
- **Start Date/Time** is the date and time of the calculation / audit read.
- **End Date/Time** is the date and time of the calculation / audit read.
- **Final Quantity** is the quantity of the calculation / audit read. This would typically contain the amount that was billed during the **Start Date/Time** through the **End Date/Time** for the **Final Unit of Measure**, **Final Time of Use** and **Final RQI**.
- **Unit of Measure** is the initial unit of measure of the calculation / audit read (if any).
- **Time of Use** is the initial time-of-use code of the calculation / audit read.
- **RQI** is the initial rate quantity identifier of the calculation / audit read.
- **Start Quantity** is the “start” quantity (if any). This value would typically only be displayed if the calculation / audit read is subtractive (i.e., you must subtract the end quantity from the start quantity to determine billable quantity).
- **End Quantity** is the “end” quantity (if any). This value would typically only be displayed if the calculation / audit read is subtractive (i.e., you must subtract the end quantity from the start quantity to determine billable quantity).
- **Quantity** is the resulting measured quantity (if any). This value would typically only be displayed if the calculation / audit read is subtractive (i.e., you must subtract the end quantity from the start quantity to determine billable quantity). It should contain the difference between the **Start Quantity** and the **End Quantity**.
- **Constant** is the constant of the measuring device that was multiplied by the **Quantity** to derive the calculation / audit read’s **Final Quantity**.
- **How To Use** is a code that indicates if the “calculation / audit” read was considered to be additive, subtractive, peak or check.
- **Use Percent** is the percentage of the total quantity that was billed.
- **Meas Peak Qty** indicates if the unit of measure is one that measures a peak amount.
- **Multiplier** is the ratio of the **Final Quantity** and the **Quantity**. This value only appears if a non-zero value exists in both of these fields.
- **Sequence** is the sequence number (if any).

This information may be overridden on a bill segment in the **Error** or **Freezable** state. Insert one row in the calculation / audit details grid (click the + button) for each entry you need to add. You may also information populated by the system when it initially generated the bill by simply overwriting the information.

**Warning!** The proper way to fix the calculation/audit details on a bill segment is to correct the cause of the error. Overriding the calculation/audit details on a bill segment is a last resort that should only be used when you can’t fix the cause of the problem.

Refer to Calculation / Audit Read Details for more information about this section.

## Bill Segment - Calc Lines

The Description of Page section below simply describes the fields on this page. Refer to [How To Cancel / Rebill A Bill Segment](#) for instructions describing how to regenerate this information if the bill segment is incorrect.

The Calc Lines page contains information about the bill calculation lines produced when the system generates the bill segment. These lines are the source of the details printed on the taxpayer's bill. Open **Financial, Bill Segment, Calc Lines** to view this information.

Most obligations use a rate to calculate the information that appears on this page. Refer to [How Rates Affect The Information On Bill Segments](#) for more information.

### Description of Page

**Bill Seg Info** is a concatenation of the bill segment's division, obligation type, status, bill period and amount. **Bill Segment ID** is the system-assigned unique identifier of the bill segment.

The Bill Segment Header (**Bill Seg Hdr**) scroll contains one entry for every version of the rate that was used to calculate the bill segment's lines. Because most bill segments use a single rate version, you typically only see one entry in the scroll.

The following information is displayed above the bill calculation line grid.

- **Sequence** will be 1 unless multiple versions of the rate were in effect during the bill period. If there was more than one effective rate version, there will be a separate set of calc lines for each version applied. The lines that apply to the first part of the bill period would have a Sequence of 1, the lines for the next part of the bill period would have a Sequence of 2 and so on.
- **Start Date – End Date** is the portion of the bill segment period that the calculation details apply.
- **Amount** is the sum of the bill calculation lines in the grid.
- **Desc on Bill** is summary information about the calculation details that will be printed on the taxpayer's bill.
- **Rate Version** is the ID of the rate version and its effective date. This information only appears if a rate was used to calculate the charges.

The calculation lines grid is a snapshot of how the system calculated the bill segment amount. One row exists for every calculation involved in this process. This information is for audit purposes only and may not be modified. The following information is displayed in the grid:

- A drill down button appears in the **Calc Line Char** column if the bill calculation line has characteristics. The Char column only appears if at least one of the calculation lines has a characteristic.
- **Sequence** is the system-assigned unique identifier of the calculation detail row.
- **Description on Bill** is the information about the bill line that appears on the taxpayer's bill.

- **Calculated Amount** is the calculated amount associated with the bill line.
- The **Print** switch controls whether information about this line will print on the taxpayer's bill.
- The **Appears in Summary** switch defines if this line's amount also appears on a summary line. This switch plays a part at bill print time – those lines that appear in a summary print in the left dollar column, those that don't appear in a summary print in the right dollar column.

The system automatically populates this switch as follows:

- Bill segments created by applying a rate have this switch turned on if the corresponding rate component is summarized on a summary rate component.
- Bill segments created from billable charges use the value of this switch as defined on the billable charge. Refer to [Maintaining Billable Charges](#) for more information.
- **Unit of Measure (UOM)** is the unit of measure of the rate quantity priced on the calculation line.
- **Time of Use (TOU)** is the time-of-use code of the rate quantity priced on the calculation line.
- **RQI** is the rate quantity identifier of the rate quantity priced on the calculation line.
- **Billable Rate Quantity** is the quantity priced on the calculation line. This quantity will be different from the measured quantity if there are RQ rules in effect.
- **Base Amount** is used by calculation lines (e.g. taxes) that are cross-referenced to other calculation lines and whose value(s), therefore, depend on the amounts calculated by those other lines. The Base Amount shows the total amount derived from the cross-referenced line(s) that the current line then used to calculate its billed amount.
- **Rate Component Sequence** refers to the sequence number of the rate component on the applicable rate version that was used to calculate the line.
- **Measures Peak Qty** is checked if the UOM priced on the calculation line is used to measure a peak quantity.
- **Exempt Amount** is the amount of the calculated charge that the taxpayer doesn't have to pay because they are tax exempt.
- **Distribution Code** is the distribution code associated with the calculation line. This distribution code is used to build the general ledger details on the bill segment's financial transaction.
- **Description** describes the characteristic value that was used when the line's amount was calculated. This information is only displayed if the line was calculated using a rate factor (because only rate factors use characteristic values). Refer to [An Illustration Of A Rate Factor And Its Characteristics](#) for more information.

## Bill Segment - Financial Details

The Description of Page section that appears below simply describes the fields on this page. Refer to [How To Cancel / Rebill A Bill Segment](#) for instructions describing how to regenerate this information if it is incorrect.

The Financial Details page contains information about the financial effects of the bill segment. Open **Financial, Bill Segment, Financial Details** to view this information.

**Note.** Refer to [The Source Of GL Accounts On A Bill Segment's Financial Transaction](#) for how the system compiles this information.

### Description of Page

**Bill Seg Info** is a concatenation of the bill segment's division, obligation type, status, bill period and amount. **Bill Segment ID** is the system-assigned unique identifier of the bill segment.

The **Bill Segment FT** scroll contains one entry for every financial transaction (FT) associated with the bill segment. All bill segments have an FT that shows the impact of the bill segment on the general ledger. A bill segment may have an additional FT (for a maximum of two) if it is cancelled. The second FT shows the impact of the cancellation on the general ledger.

The following financial transaction information is displayed above the general ledger distribution grid.

- **FT Type** displays the type of financial transaction (FT).
- **Show on bill** indicates if information about the FT appears on the taxpayer's bill.
- **FT ID** is the system-assigned unique identifier of the FT. Click the go to button to transfer to the [Financial Transaction - Main](#) page. On this page you can change certain aspects of the FT in question. Refer to [How To Remove Unwanted Adjustments \(or Payments\) From A Completed Bill](#) for a description of how to perform common maintenance functions.
- The posted by information that appears adjacent to the FT ID shows the date on which the FT was frozen and the user ID of the user who performed the operation.
- The "Linked to bill on" message appears when the financial transaction is linked to a bill. A FT is linked to a bill the next time a bill is completed for the account. Therefore, if a user cancels a bill segment, the FT associated with the cancellation will not be linked to a bill until the next bill is completed (although the financial effect of the cancellation takes place immediately). Click the adjacent go to button to transfer to the bill on which the FT appears.
- **Current Amount** contains the FT's effect on the obligation's current balance.
- **Payoff Amount** contains the FT's effect on the obligation's payoff balance. This field is only visible when the payoff amount differs from the current amount.
- **Accounting Date** is the date the system uses to determine the FT's accounting period in your general ledger.
- **Effective Date** is the date the FT starts aging.

The journal details grid is a snapshot of the distribution codes used to generate the GL accounts that are affected by the FT. This information is for audit purposes only and may not be modified. The following information is displayed in the grid:

- **Sequence** is the system-assigned unique identifier of the distribution line.
- **Distribution Code** is the distribution code used to generate the GL account affected by the distribution line. Drill over to the FT if you need to see the actual GL account.
- **Amount** is the amount debited / credited (credits are shown as negative values).
- **Statistics Code** and **Statistics Amount** are only specified when a statistical quantity is associated with the distribution line.

- **Description** describes the characteristic value that was used when the line's amount was calculated. This information is only displayed if the distribution line was derived from bill calculation lines that were calculated using a rate factor (because only rate factor's use characteristic values). Refer to [An Illustration Of A Rate Factor And Its Characteristics](#) for more information.
- If you keep the default installation option where [fund accounting](#) is **practiced**, the description of the **Fund** associated with this distribution code is displayed.

## Bill Segment - Bill Segment Messages

The Messages page is a grid containing one row for every message that appears on the bill segment. Open **Financial, Bill Segment, Bill Segment Messages** to view this information.

**No messages until completion.** A bill segment has no messages until it is completed (unless you insert them manually). At completion time, the system assembles messages from the various message sources. Refer to [The Source Of Bill Messages](#) for information about these sources.

**The bill may also have messages.** Be aware that only obligation-oriented messages are linked to a bill segment. There may also be account-oriented messages linked to the bill. Refer to [Bill - Bill Messages](#) for information about the page on which account-oriented messages are displayed. Refer to [The Source Of Bill Messages](#) for information about the various message sources and whether each is linked to a bill or a bill segment.

### Description of Page

**Bill Seg Info** is a concatenation of the bill segment's division, obligation type, status, bill period and amount. **Bill Segment ID** is the system-assigned unique identifier of the bill segment.

The following columns are displayed in the grid:

- **Msg Code** is the code that identifies the bill message.
- **Message On Bill** is the message associated with the code.
- **Priority** is the bill messages priority (on the printed bill).
- **Insert Code** defines if the bill message causes an insert in the envelope.

## Multi Cancel/Rebill

The Multi-Cancel/Rebill transaction is used to cancel / rebill (and freeze) one or more bill segments linked to an obligation.

**Note.** You would typically use this transaction when something has been wrong for an extended period of time. For example, if the taxpayer was assigned the wrong rate from the beginning, you would correct the taxpayer's rate using [Obligation – Rate Info](#) and then use this transaction to cancel / rebill the taxpayer's historical bill segments.

The following topics describe how to use this transaction.

## Contents

- [Multi Cancel/Rebill - Main](#)
- [Multi Cancel/Rebill - Graph](#)

## Multi Cancel/Rebill - Main

The main page is used to select the bill segments to be cancel/rebilled and to initiate the cancel/rebill request. Open **Financial, Multi-Cancel/Rebill** to use this transaction.

### Description of Page

This page contains all non-cancelled bill segments associated with a given **Obligation ID**. The obligation's **Obligation Info**, **Account ID** (and name), characteristic **Location** (and address), **Current Balance** and **Payoff Balance** are displayed at the top of page.

When the page first opens, the grid is populated with ALL non-cancelled bill segments displayed in reverse chronology order. All **frozen** bill segments are pre-selected for cancel/rebill processing. The following information appears in the grid:

- **Selected Switch.** If this switch is turned on, the bill segment will be cancel / rebilled when the **Cancel/Rebill/Freeze** button is clicked. You can use the mouse to toggle the value of this switch on a specific bill segment. Alternatively, you can click the **Select All** button or **Unselect All** button to toggle the switch for all **frozen** bill segments. The checkbox is disabled if the bill segment's bill is written off. Note: the total number of selected bill segments is shown above the grid for confirmation purposes.
- **Start Date.** This is the first day of the bill segment.
- **End Date.** This is the last day of the bill segment.
- **Create Date/Time.** This is the day/time the bill segment was created.
- **Amount.** This is the payoff amount of the bill segment.
- **Original Amount.** This column only contains a value if the bill segment is a rebill of another bill segment. In this situation, one of the following values will be displayed:
  - If the amount of the new bill segment differs from the amount of the original bill segment, the amount of the original bill segment will be displayed.
  - If the amount of the new bill segment equals the amount of the original bill segment, the word **No change** will be displayed.
- **Bill Segment Information.** This is the standard bill segment summary information.

**Sorting By Values In The Grid.** Just like all grids in the system, you can click on a column heading to resort the grid by the value of the respective column. This might be useful to sort the bill segments in **Amount** order.

After selecting the bill segments to be cancel/rebilled, click the **Cancel/Rebill/Freeze** button.

Please specify the following information and then click **Calculate** to cancel the selected bill segment and create new bill segments:

- **Cancel Reason** defines why the bill segment(s) are being canceled.

- **Accounting Date** is used to define the financial period to which the canceled and new bill segments' financial transactions will be booked.
- Choose **Use Cut off Date** as **Billing Option**.
- If you'd prefer to have the system use the details snapshot on the original bill segments, turn on **Use Old RQ**.

**Warning!** If the cause of the rebill is an incorrect rate and the new rate requires rate quantities that were not calculated when the bill was originally calculated, do not turn on **Use Old RQ**. Why? Because you want the system to derive new rate quantities during the rebill process.

**Default note.** The Accounting Date is defaulted to the current date and the Use Old RQ switch is turned on.

After specifying the above information, the system cancel / rebills each segment in chronological order (i.e., earlier segments will be cancel / rebilled before later segments). The new amount of each bill segment is shown in the **Original Amount** column. You can navigate to the [Graph](#) page to graphically view the financial ramifications of the cancel/rebills.

**Errors.** It's possible that bill segment errors will occur when you cancel / rebill the bill segments. For example, if you change the obligation's rate to a rate that is no longer effective on the bill segment period, a bill segment error will be generated. If this occurs, you must go to [Bill Segment – Main](#) for the **pending cancel** bill segment and use the **Undo** action to remove the offending bill segment (and restore the original segment to the **Frozen** state). Alternatively, you can drill down on the **Error** bill segment and use the **Regenerate** action after correcting the cause of the problem.

**There is no Undo.** Unlike cancel / rebills performed on [Bill Segment – Main](#), there is no Undo action. This is because the multi-cancel/rebill transaction freezes the newly created bill segments after it cancel/rebills; whereas the bill segment transaction lets you examine the new bill segment before you freeze it. If you made a mistake, simply correct the cause of the mistake and then cancel / rebill again. The erroneous financial transactions will be automatically suppressed on the next bill produced for the taxpayer.

## Multi Cancel/Rebill - Graph

This page contains a graph that shows the rebilled and original payoff amounts for an obligation's non-cancelled bill segments. Open **Financial, Multi Cancel/Rebill** and navigate to the **Graph** page to view this information.

### Description of Page

This page contains a graph that shows the payoff amounts for an obligation's non-cancelled bill segments (represented by the blue bars in the graph). If a bill segment is a rebill of another bill segment, the payoff amount of the original bill segment is displayed on a red-colored bar adjacent to the respective blue bar.



**Click a bar.** You can click on a bar to drill down to a bill segment.

## Obligation Billing History

Open **Financial Query, Obligation Billing History** to view a history of all bill segments produced for an obligation.

### Description of Page

The **Account ID** is displayed with the account's name adjacent.

The **Obligation Information** shows a summary of information about the obligation. The **Obligation ID** is the unique identifier of this obligation.

One row is displayed for every bill segment ever produced for the selected obligation.

The following information is displayed for each bill segment:

- The **Start Date** and **End Date** of the bill segment.
- The number of **Days** in the bill segment period.
- The **Status** of the bill segment
- The **Current** and **Payoff Amounts** of the bill segment.
- The **UOM** (Unit of Measure) of the rate quantity designated as the "graph UOM" for the obligation's obligation type.
- The total amount of the service **Billable Rate Quantity** that was billed.
- The **Average Daily Rate Quantity** is the **Billable Rate Quantity** divided by the numbers of days of service.

If you need to see more detailed information about the bill segment, click the go to button to transfer to the bill segment page.

## Bill Exception

If errors are detected during the billing process, it may cause a record to be written to the bill exception table with a message indicating the nature of the severe error.

To view the messages associated with the exception records, schedule the [TD-BIERR](#) background process. This process generates a To Do entry for every record in the bill exception table.

Refer to [How to Correct a Bill That's In Error](#) for instructions describing how to correct a bill.

## Bill Segment Exception

If errors are detected during the billing process, it may cause a record to be written to the bill segment exception table with a message indicating the nature of the severe error.



To view the messages associated with the exception records, schedule the [TD-BSERR](#) background process. This process generates a To Do entry for every record in the bill segment exception table.

Refer to [How To Correct A Bill Segment That's In Error](#) for instructions describing how to correct a bill segment.

## Maintaining Billable Charges

You create a billable charge whenever a taxpayer should be charged for a service that occurs outside the normal course of business. For example, if a taxpayer requires a review of their property assessment, you may charge them an administration fee.

**Billable charges can be uploaded from an external system.** Refer to [Uploading Billable Charges](#) and [Using Billable Charges For Pass Through Billing](#) for more information.

A billable charge must refer to an obligation. This obligation behaves just like any other obligation:

- **Bill segments are created for the obligation.** Whenever billing is performed for an account with billable charge obligations, the system creates a bill segment for each billable charge. This means that if an obligation has many unbilled billable charges, it will have many bill segments.
- **Payments are distributed to the obligation.** Payments made by an account are distributed to its billable charge obligations just like any other obligation.
- **Overdue debt is monitored.** The collections process monitors billable charge obligations for overdue debt and responds accordingly when overdue debt is detected.
- This obligation must reference an obligation type that has a Special Role Flag of **Billable Charge**. If such an obligation does not already exist for the account, one must be created before you can levy a billable charge. See “For more information” at the end of this section for links to other sections that will help you with this process.

Billable charge templates exist to minimize the effort required to create a billable charge for a taxpayer. A billable charge template contains the standard bill lines, amounts and distribution codes used to charge for a one-off charge.

A user may override the information on the template when the billable charge is created.

**Templates aren't required.** A billable charge can be created without a template for a truly unexpected charge. If you don't use a template, you'll have to enter the bill information manually.

Refer to [Setting Up Billable Charge Templates](#) for more information.

**Adding taxes and other calculation lines to a billable charge.** Refer to [Billable Charge – RQ Details](#) for information about how to have the system calculate tax lines and append them to the billable charge.

## Contents

- [Billable Charge - Main](#)
- [Billable Charge - Line Characteristics](#)
- [Billable Charge - RQ Details](#)
- [Billable Charge - Calculation Details](#)

## Billable Charge - Main

Open **Financial**, **Billable Charge** to add or update a billable charge.

### Description of Page

The **Billable Charge ID** is system generated. The **Account ID** of the account responsible for the charge is displayed at the top.

Use **Obligation ID** to define the ID of the billable charge obligation that will hold the billable charge's debt. The **Account ID** for the obligation is displayed for information purposes.

Use **Start Date** and **End Date** to define the charge's bill period.

**Installment billable charges.** You can create many future dated billable charges to implement unusual installment plans. For example, if you want to bill a taxpayer \$250 today, \$150 next month, and \$1000 three months from now; you could create three billable charges with a date that reflects the earliest bill date. When a bill is produced for the account, the system will create a bill segment for every billable charge whose start date is on or before the current date.

If you want to create the billable charges from an existing template, enter the ID of the **Billable Charge Template** from which the billable charge information defaults. If your [obligation type](#) indicates a billable charge template, that value will be defaulted.

Use **Description on Bill** to define the verbiage that should print on the taxpayer's bill above the line item details.

**Billable Charge Status** defines the state of the billable charges. Possible values are **Billable** and **Canceled**. Click the **Cancel** button to cancel a billable charge.

**Bill Segment ID** is the ID of the bill segment on which the billable charge's charge details appear. This information is only populated AFTER the billable charge has been swept onto a bill.

The **Total Bill Amount** displays the sum of all billable charge lines except those that are **Memo Only**. The **Total Line Amount** displays the sum of all billable charge lines except those that **Appear In Summary**.

The information in the grid defines the line item details associated with the billable charge. Click the drill down button to navigate to the line characteristics. The following fields are required for each line:

<b>Sequence</b>	Sequence controls the order in which the line items appear on the bill segment.
<b>Description on Bill</b>	Specify the verbiage to print on the bill for the line item.
<b>Charge Amount</b>	Specify the amount to charge for the line item.
<b>Show on Bill</b>	Turn this switch on if the line item should appear on the taxpayer's printed bill. It would be very unusual for this switch to be off.

**Appears in Summary**

Turn this switch on when the amount associated with this line also appears in a summary line. This switch plays a part at bill print time - those lines that appear in a summary print in the left dollar column, those that don't appear in a summary print in the right dollar column. Also note, those lines that appear in a summary line are not included in the total amount associated with the billable charge.

**Memo Only, No GL**

Turn this switch on if the billable charge line does not contribute to the total amount due (e.g., this switch would be on for "information only" lines).

**Distribution Code**

Specify the distribution code that identifies the GL account associated with this line item. The description of the code appears adjacent.

For more information about billing billable charges, refer to [How To Create An Ad-hoc Bill](#). For more information about billable charge templates, refer to [Setting Up Billable Charge Templates](#).

## Billable Charge - Line Characteristics

Characteristics may be associated with billable charge lines. Typically, this is done to categorize billable charge lines for reporting purposes.

Open **Financial, Billable Charge, Line Characteristics** to maintain characteristics that are associated with billable charge.

**Description of Page**

The **Billable Charge Line** scroll defines the billable charge line to which you wish to assign characteristic values. The following fields display:

<b>Characteristic Type</b>	The type of characteristic.
<b>Characteristics Value</b>	The value of the characteristic.

**Note.** You can only choose characteristic types defined as permissible on a billable charge line. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

## Billable Charge - RQ Details

Optionally, you may specify rate quantities associated with a billable charge. These are most commonly used when you want the system to calculate charges *in addition to the billable charge lines* (e.g., adding tax to the billable charge lines).

**The obligation must specify a rate.** If you want the system to calculate additional charges based on the rate quantities defined on this page, the related billable charge obligation must reference a rate. This rate must have rate components that calculate charges based on the rate quantities specified on this page. When the obligation is next billed, the system will sweep on the billable charges and then call the rate and pass it the rate quantities defined on this page. The rate will calculate the additional charges and these will be appended to the bill segment.

Open **Financial, Billable Charge, RQ Details** to maintain a billable charges rate quantities.

### Description of Page

The following fields display:

<b>Sequence</b>	Specify the sequence number of the RQ.
<b>Unit of Measure</b>	Select the unit of measure of this RQ. One or more of <b>Unit of Measure, Time of Use, or Rate Quantity Identifier</b> must be selected.
<b>Time of Use</b>	Select the time of use period.
<b>Rate Quantity Identifier</b>	Select the RQ identifier.
<b>Rate Quantity</b>	Specify the number of units of this rate quantity.

## Billable Charge - Calculation Details

Optionally, you may specify calculation details associated with a billable charge. These are most commonly used when you need to print the details that were used to calculate the bill lines that appear on the main tab.

Open **Financial, Billable Charge, Calculation Details** to maintain a billable charge's rate quantities.

### Description of Page

The grid is a snapshot of the details that were billed using the bill lines shown on the main tab. One row exists for every calculation detail that should be swept onto the related bill segment when the system creates a bill segment for this billable charge. The following information is displayed in the grid:

- **Final Unit of Measure** is the final unit of measure of the calculation / audit read.
- **Final Time of Use** is the final time-of-use code of the calculation / audit read.
- **Final RQI** is the final rate quantity identifier of the calculation / audit read.
- **Final Quantity** is the quantity of the calculation / audit read. This would typically contain the amount that was billed during the **Start Date/Time** through the **End Date/Time** for the **Final Unit of Measure, Final Time of Use** and **Final RQI**.
- **Sequence** is the sequence number (if any).
- **Unit of Measure** is the initial unit of measure of the calculation / audit read (if any).
- **Time of Use** is the initial time-of-use code of the calculation / audit read.

- **RQI** is the initial rate quantity identifier of the calculation / audit read.
- **Start Quantity** is the “start” quantity (if any). This value would typically only be displayed if the calculation / audit read is subtractive (i.e., you must subtract the end quantity from the start quantity to determine billable quantity).
- **End Quantity** is the “end” quantity (if any). This value would typically only be displayed if the calculation / audit read is subtractive (i.e., you must subtract the end quantity from the start quantity to determine billable quantity).
- **Quantity** is the resulting measured quantity (if any). This value would typically only be displayed if the calculation / audit read is subtractive (i.e., you must subtract the end quantity from the start quantity to determine billable quantity). It should contain the difference between the **Start Quantity** and the **End Quantity**.
- **Constant** is the constant of the measuring device that was multiplied by the **Quantity** to derive the calculation / audit read’s **Final Quantity**.
- **How To Use** is a code that indicates if the “calculation / audit” read is considered to be additive, subtractive, peak or check.
- **Use Percent** is the percentage of the total quantity that was billed.
- **Measures Peak Qty** indicates if the unit of measure is one that measures a peak amount.
- **Start Date/Time** is the date and time of the calculation / audit read.
- **End Date/Time** is the date and time of the calculation / audit read.
- **Multiplier** is the ratio of the **Final Quantity** and the **Quantity**. This value only appears if a non-zero value exists in both of these fields.

**Note.** The multiplier is calculated by dividing the Final Quantity by the Quantity (if the Quantity is 0, then the result is 0). This is not stored on the database. There is a chance that the value is not precise due to round-off.

## Uploading Billable Charges

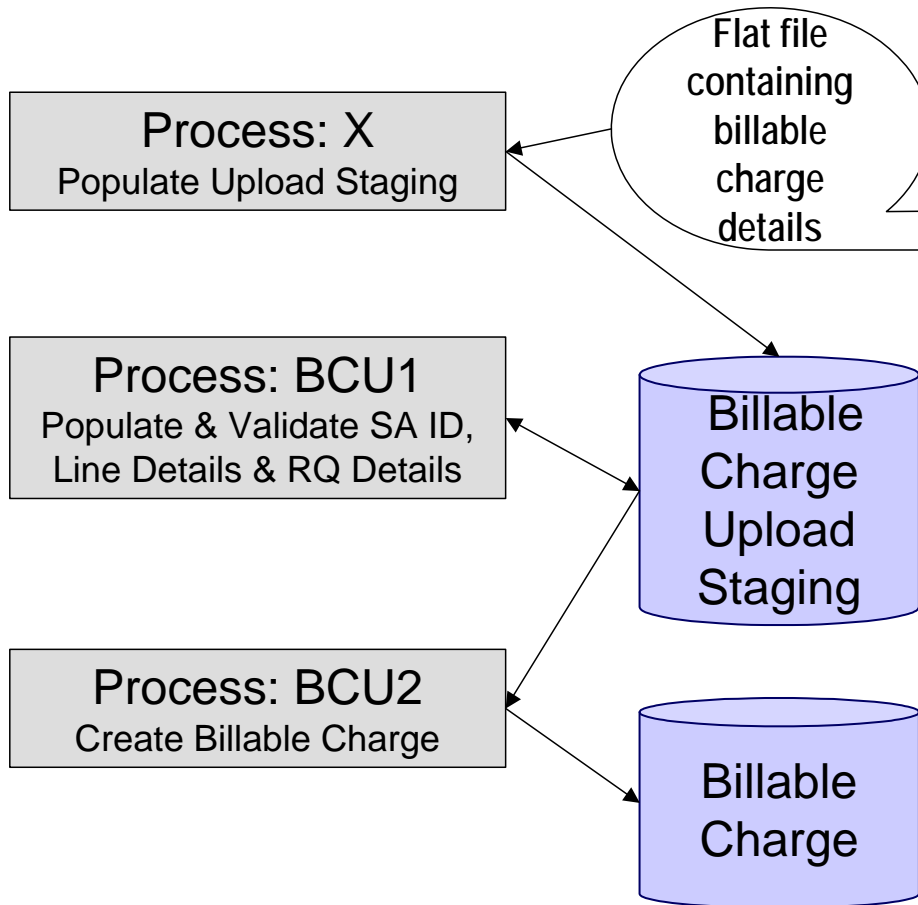
This section describes how the system uploads billable charges from an external source. This mechanism is used when you need to ["pass through" charges](#) that have been calculated by a third party but appear on your bill.

### Contents

- [Billable Charge Upload Background Processes](#)
- [BCUP-PRG - Purge Billable Charge Upload Objects](#)
- [Billable Charge Upload Staging](#)
- [Billable Charge Upload Exception](#)

## Billable Charge Upload Background Processes

The following diagram illustrates the processes involved in the uploading of billable charges into the system.



The topics in this section describe how these processes work.

#### Contents

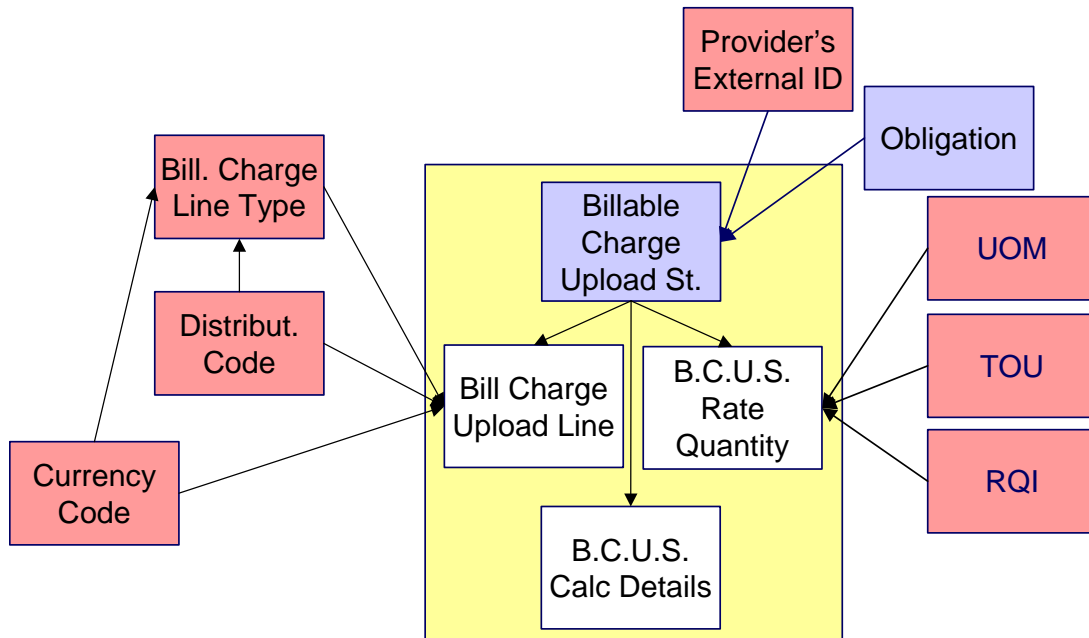
[Process X - Populate BC Upload Staging](#)

[BCU1 - Validate & Populate Billable Charge Upload Staging](#)

[BCU2 - Create Billable Charge](#)

#### Process X - Populate BC Upload Staging

Process X refers to the mechanism used by your organization to populate the various staging tables (shown in the yellow section of the following ERD).



The topics in this section describe each of these tables.

### Contents

- [Billable Charge Upload Staging Layout](#)
- [Billable Charge Line Upload Staging Layout](#)
- [Billable Charge Line Characteristic Upload Staging Layout](#)
- [Billable Charge Rate Quantity Upload Staging Layout](#)
- [Billable Charge Calculation Details Upload Staging Layout](#)

### Billable Charge Upload Staging Layout

You must create an upload staging record for each billable charge. The name of this table is [CI\\_BCHG\\_UP](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
BCHG_UP_ID	30	Y	Char	This is the unique identifier of the record. This value does NOT have to be a random number, but it does need to be unique. If your process that inserts records on this table is capable of calling a COBOL routine, call CIPCKEYG and it will supply a 12 digit random number for you.
BCHG_UP_STAT_FLG	2	Y	Char	This must be set to <i>P</i> for <i>Pending</i> .
NT_XID_CD	30	N	Char	Leave this column blank.
SA_EXT_REF_ID	36	N	Varchar2	Leave this column blank.
BCHG_EXT_REF_ID	50	N	Varchar2	This is the identifier of the billable charge in the sender's system.
CRE_DTTM	26	N	Date/Time	The date and time the upload staging row was inserted (this can

				be used for audit purposes).
START_DT	10	Y	Date	The start date of the period encompassed by the billable charge.
END_DT	10	Y	Date	The end date of the billable charge period.
DESCR_ON_BILL	80	Y	Varchar2	This is the description that should prefix the charges on the printed bill.
SA_ID	10	See note	Char	This must correspond with an obligation ID of a billable charge obligation.
BILLABLE_CHG_ID	12	N	Char	Leave this column blank. It will be assigned by the system when it creates a billable charge record.

### Billable Charge Line Upload Staging Layout

You must create a billable charge line upload record for each line to be uploaded. The name of this table is [CI\\_BCHG\\_LINE\\_UP](#). The following table describes each column on this table.

**Defaulting from Billable Charge Upload Line Type (BCHG\_UP\_XTYP).** Please pay special attention to the **Comments** column below as many fields, if left blank, will default from each record's billable charge upload line type (BCHG\_UP\_XTYP).

Column Name	Length	Req'd	Data Type	Comments
BCHG_UP_ID	30	Y	Char	This is the foreign key to the billable charge upload staging record.
LINE_SEQ	3	Y	N	This is the unique line number of the billable charge line. This value must be unique within the billable charge.
DESCR_ON_BILL	80	Y	Varchar2	This is the description that will be printed on the bill for the line.
CHARGE_AMT	13.2	N	N	This is the amount associated with the billable charge line.
BCHG_LINE_XID	20	N	Varchar2	This is the unique identifier of the billable charge line in the sender's system.
BCHG_UP_XTYPE	30	N	Char	This is the type of billable charge line. This is a foreign key reference to the billable charge line type control table. This control table is used to populate the remaining fields (if they are blank). If this field is blank, the remaining fields must be specified as they cannot be defaulted from the billable charge



				line type control table. Refer to <a href="#">Bill Charge Line Type</a> for more information.
CURRENCY_CD	3	See note	Char	This is a foreign key reference to the currency code of the billable charge line. This field must be specified if CHARGE_AMT is non-zero.
SHOW_ON_BILL_SW	1	See note	Char	This should be <b>Y</b> if the line should appear on the taxpayer's printed bill. This should be <b>N</b> if the line should not appear on the taxpayer's printed bill. If left blank, this value will be populated from the line's BCHG_UP_XTYPE.
APP_IN_SUMM_SW	1	See note	Char	This should be <b>Y</b> if the line appears in a summary line. This should be <b>N</b> if the line does not appear in a summary line. This switch plays a part at bill print time – those lines that appear in a summary print in the left dollar column, those that don't appear in a summary print in the right dollar column. If left blank, this value will be populated from the line's BCHG_UP_XTYPE.
MEMO_SW	1	See note	Char	This should be <b>Y</b> if the line does NOT affect the general ledger. This should be <b>N</b> if the line affects the general ledger. If left blank, this value will be populated from the line's BCHG_UP_XTYPE.
DST_ID	10	See note	Char	If the line affects the general ledger, the set ID (the previous field) and the distribution code of the GL account must be specified. If left blank, this value will be populated from the line's BCHG_UP_XTYPE.

### Billable Charge Line Characteristic Upload Staging Layout

If you want to upload characteristics on your billable charge lines, you must create a billable charge line characteristic upload record for each characteristic on each line. The name of this table is [CI\\_B\\_LN\\_UP\\_CHAR](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
BCHG_UP_ID	30	Y	Char	This is the foreign key to the billable charge upload staging record.
LINE_SEQ	3	Y	N	This is the foreign key to the billable charge line upload staging record

CHAR_TYPE_CD	8	Y	Char	This is the unique identifier of the billable charge line characteristics. This value must be unique within the billable charge line characteristics.
CHAR_VAL	16	N	Char	A characteristic value must be supplied if the ad hoc switch on characteristic type is <i>N</i> . Otherwise, it is not allowed.
ADHOC_CHAR_VAL	30	N	Varchar2	An ad hoc characteristic value must be supplied if the ad hoc switch on characteristic type is <i>Y</i> . Otherwise, it is not allowed.

### Billable Charge Rate Quantity Upload Staging Layout

You must create a billable charge rate quantity upload record for each rate quantity to be uploaded. The name of this table is [CI BCHG UP SQ](#). The following table describes each column on this table. Refer to ["pass through" charges](#) for more information about how this type of information is used.

Column Name	Length	Req'd	Data Type	Comments
BCHG_UP_ID	30	Y	Char	This is the foreign key to the billable charge upload staging record.
SEQ_NUM	3	Y	N	Sequence of the rate quantity within the Billable Charge Upload
UOM_CD	4	N	Char	The unit of measure of the Billable Charge Upload RQ row. If specified, it must reference a valid value on CI_UOM. Note, at least one of the following fields should be specified: SQI_CD, TOU_CD, UOM_CD.
TOU_CD	8	N	Char	The time of use of the Billable Charge Upload RQ row. If specified, it must reference a valid value on CI_TOU. Note, at least one of the following fields should be specified: SQI_CD, TOU_CD, UOM_CD.
SQI_CD	8	N	Char	The rate quantity identifier of the Billable Charge Upload RQ row. If specified, it must reference a valid value on CI_SQI. Note, at least one of the following fields should be specified: SQI_CD, TOU_CD, UOM_CD.
SVC_QTY	12.6	N	N	Quantity of the rate used for the UOM / TOU / RQI combination

**Billable Charge Calculation Details Upload Staging Layout**

You must create a billable charge read details upload record for each calculation detail to be uploaded. The name of this table is [CI\\_BCHG\\_UP\\_READ](#). The following table describes each column on this table. Refer to [“pass through” charges](#) for more information about how this type of information is used.

Column Name	Length	Req'd	Data Type	Comments
BCHG_UP_ID	30	Y	Char	This is the foreign key to the billable charge upload staging record.
SP_ID	10	Y	Char	Leave this column blank. It is not used.
SEQNO	5	Y	Number	This should be a value greater than zero.
END_READ_DTTM	26	Y	DateTime	This is the date and time of the end quantity
END_REG_READ_ID	12	N	Char	Leave this column blank. It is not used.
END_REG_READING	9.6	Y	Number	This is the end quantity.
FINAL_REG_QTY	12.6	Y	Number	This is the final quantity
FINAL_SQI	8	N	Char	The rate quantity identifier of the FINAL_REG_QTY. If specified, it must reference a valid value on CI_SQI. Note, at least one of the following fields should be specified: FINAL_SQI, FINAL_UOM_CD.
FINAL_TOU_CD	8	N	Char	The time-of-use code of the FINAL_REG_QTY. If specified, it must reference a valid value on CI_TOU.
FINAL_UOM_CD	4	N	Char	The unit of measure of the FINAL_REG_QTY. If specified, it must reference a valid value on CI_UOM. Note, at least one of the following fields should be specified: FINAL_SQI, FINAL_UOM_CD.
HOW_TO_USE_FLG	2	Y	Char	See the field description in the data dictionary for the valid values ( <a href="#">CI_BCHG_UP_READ</a> ).
MSR_PEAK_QTY_SW	1	Y	Char	Y or N
MSR_QTY	12.6	Y	Number	Quantity. This is the difference between end quantity and start quantity.
REG_CONST	6.6	Y	Number	This is the constant value used to adjust the quantity. Set this to a value of 1 if it is not known.

SQL_CD	8	N		The rate quantity identifier of end quantity and start quantity. If specified, it must reference a valid value on CI_SQL. Note, at least one of the following fields should be specified: SQL_CD, TOU_CD, UOM_CD.
START_READ_DTTM	26	Y	DateTime	This is the date and time of the start quantity.
START_REG_READ_ID	12	Y	Char	Leave this column blank. It is not used.
START_REG_READING	9.6	Y	Number	This is the start quantity
TOU_CD	8	N	Char	The time-of-use code of end quantity and start quantity. If specified, it must reference a valid value on CI_TOU.
UOM_CD	4	Y	Char	The unit of measure of end quantity and start quantity.
USAGE_FLG	2	Y	Char	Leave this column blank. It is not used.
USE_PCT	3	Y	Number	This is the percentage of the quantity that was billed (this will typically be 100).

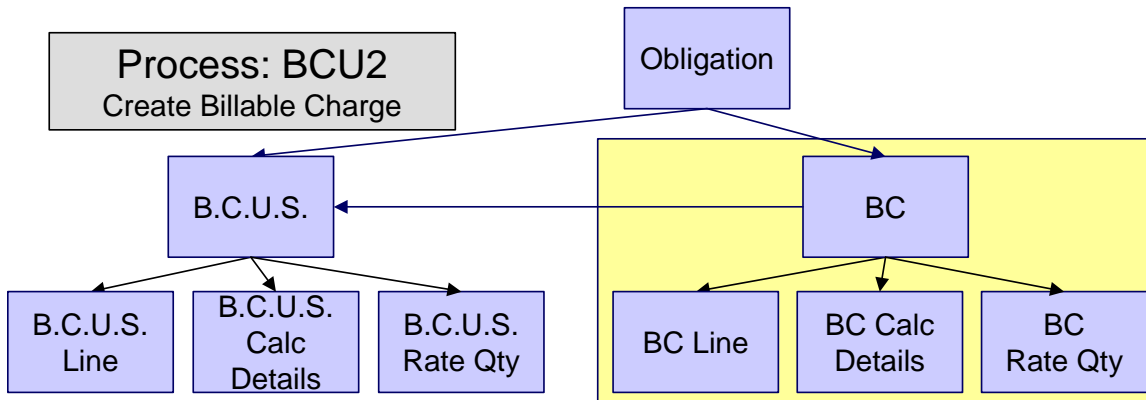
### BCU1 - Validate & Populate Billable Charge Upload Staging

This process populates various fields (e.g., the GL distribution code, memo only switch) on the billable charge upload line records from the billable charge line type specified on each respective record.

Any validation / population errors detected during this process are written to the [BC Upload Exception](#) table. You can fix errors using BC Upload Staging page (don't forget to change the record's status back to **Pending**).

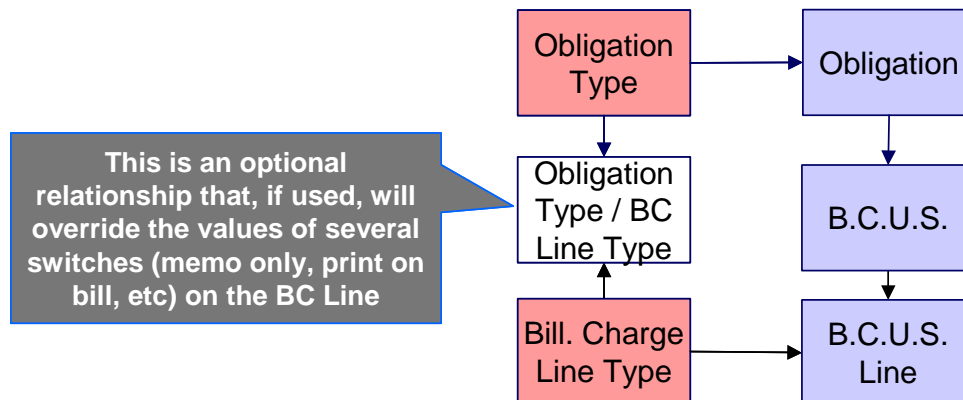
### BCU2 - Create Billable Charge

This process creates a billable charge and billable charge lines (shown in the yellow section of the following ERD) for all BC upload records in the **Pending** state.



If the billable charge's obligation is **Closed**, it will **Reactivate** it so that the billable charges will be swept onto the next bill produced for the account.

**Note.** This process will override the values of the various switches referenced on bill charge upload staging line if the respective obligation's obligation type has an override value for the line's billable charge line type. Refer to [Obligation Type – Billable Charge Overrides](#) for more information.



Any errors detected during this process are written to the [Billable Charge Upload Exception](#) table. You can fix errors using BC Upload Staging page (don't forget to change the record's status back to **Pending**).

## BCUP-PRG - Purge Billable Charge Upload Objects

**Completed** billable charge upload staging objects should be periodically purged from the system by executing the [BCUP-PRG](#) background process. This background process allows you to purge all **Completed** billable charge upload staging objects older than a given number of days.

We want to stress that there is no system constraint as to the number of **Completed** billable charge upload objects that may exist. You can retain these objects for as long as you desire. However we recommend that you periodically purge **Completed** billable charge upload objects as they exist only to satisfy auditing and reporting needs.

## Billable Charge Upload Staging

The Billable charge Upload Staging page has three purposes:

- You can view historical billable charge upload records.
- You can correct billable charge upload records that are in error.
- You can add new billable charge upload records. These will be uploaded by the billable charge upload process (although it's probably easier to just add the billable charges using the [Maintaining Billable Charges](#) page).

The topics in the section describe how to maintain this information.

### Contents

- [Billable Charge Upload - Main](#)
- [Billable Charge Upload - Lines](#)
- [Billable Charge Upload - RQ Details](#)
- [Billable Charge Upload - Calculation Details](#)

### Billable Charge Upload - Main

Open this page using **Financial, Billable Charge Upload Staging, Main**.

#### Description of Page

Refer to [Billable Charge Upload Staging](#) for more information about these fields.

**Correcting Errors.** If a Billable Charge Upload record's status is **Error**, you should correct the problem and then change its status back to **Pending**. The next time the billable charge upload process runs, it will revalidate the record and, if valid, it will create a billable charge record.

### Billable Charge Upload - Lines

Open this page using **Financial, Billable Charge Upload Staging, Lines**.

#### Description of Page

Refer to [Billable Charge Line Upload Staging](#) for more information about these fields.

### Billable Charge Upload - RQ Details

Open this page using **Financial, Billable Charge Upload Staging, RQ Details**.

#### Description of Page

Refer to [Billable Charge Rate Quantity Upload Staging](#) for more information about these fields.

## Billable Charge Upload - Calculation Details

Optionally, you may specify calculation details associated with a billable charge. These are most commonly used when you need to print the details that were used to calculate the bill lines that appear on the main tab.

Open **Financial, Billable Charge, Calculation Details** to maintain a billable charge's rate quantities.

### Description of Page

Refer to [Billable Charge Calculation Details Upload Staging](#) for more information about these fields.

## Billable Charge Upload Exception

If errors are detected during the billable charge upload process, a record is written to the billable charge upload exception table with a message indicating the nature of the severe error.

To view the messages associated with the exception records, schedule the [TD-BCUPL](#) background process. This process generates a To Do entry for every record in the billable charge upload exception table.

You can fix this error using the [Billable Charge Upload Staging](#) page and change the status of the record from **Error** to **Pending**. When the billable charge upload process next runs, it attempts to upload this record again.

# Payments

In this section, we describe how to manage your taxpayer's payments.

## Contents

- [The Big Picture of Payments](#)
- [Maintaining Payment Events](#)
- [Payment Event QuickAdd](#)
- [Payment QuickAdd](#)
- [Maintaining Payments](#)
- [How To](#)
- [Financial Transactions On A Payment](#)
- [Payment History](#)
- [Payment / Tender Search](#)
- [Payment Event Exception](#)
- [Payment Exception](#)
- [Maintaining Deposit Controls](#)
- [Maintaining Tender Controls](#)
- [Interfacing Payments From External Sources](#)

## The Big Picture of Payments

---

A payment reduces how much an account owes. The topics in this section provide background information about a variety of payment topics.

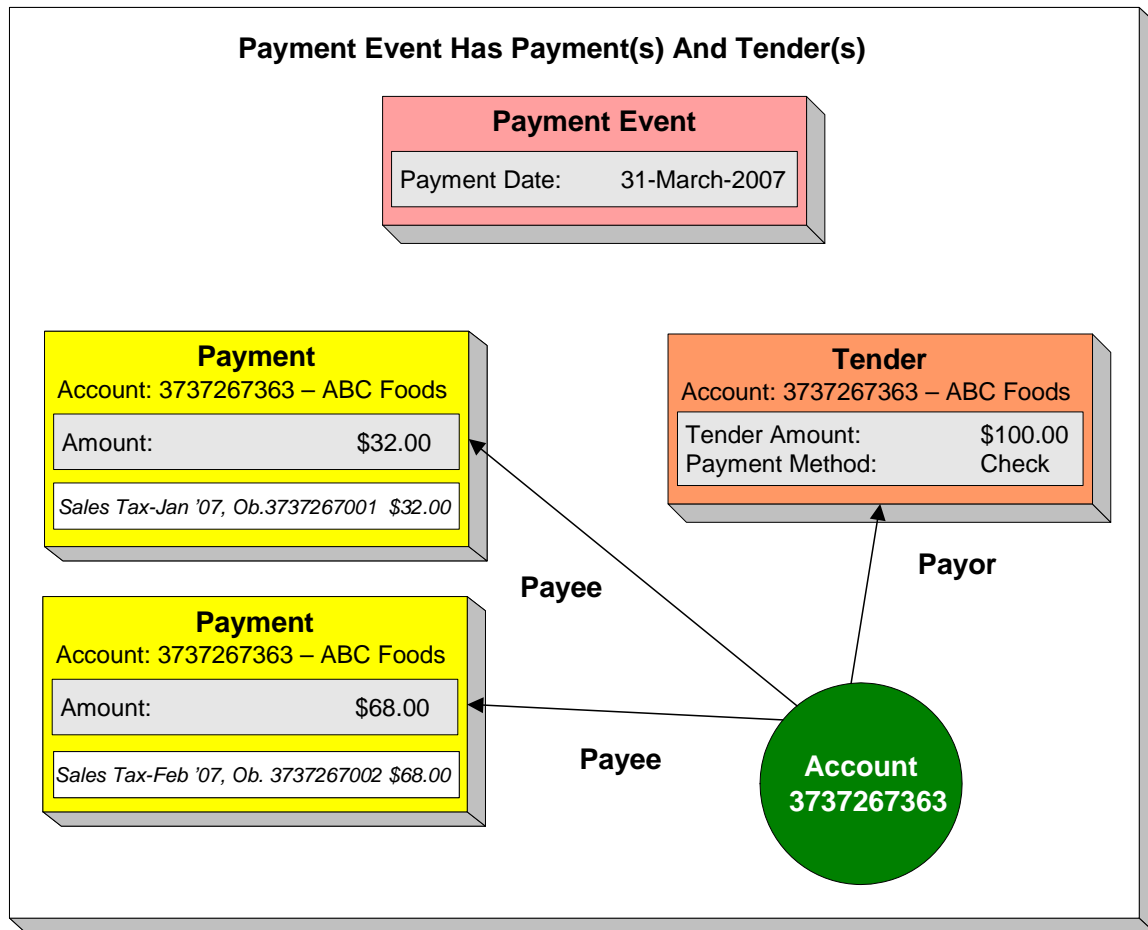
## Contents

- [A Payment Event Has Payments And Tenders](#)
- [Multiple Tenders Used To Pay For Multiple Accounts](#)
- [An Overview Of The Payment Event Creation & Allocation Process](#)
- [Payments and Penalty and Interest](#)
- [Payment Date and Effective Date for Payment Events](#)
- [Distributing A Payment Event](#)
- [Distributing A Payment Amongst An Account's Obligations](#)
- [Overpayment](#)
- [Canceling A Tender Versus Canceling A Payment](#)
- [NSF Cancellations](#)
- [Transferring A Payment](#)
- [Unbalanced Payment Events](#)
- [Tender Management and Workstation Cashiering](#)
- [Exceptions](#)
- [Payment Financial Transaction Considerations](#)
- [A Payment May Affect More Than Just Taxpayer Balances](#)
- [Automatic Payments](#)
- [Issuing A Payment Advice Instead Of Creating An Automatic Payment](#)

## A Payment Event Has Payments And Tenders

The explanation in [The Financial Big Picture](#) provides an accurate, but incomplete view of payments. The missing pieces concern *payment events* and *tenders*. The following diagram illustrates the difference between a payment event, its payment(s) and its tender(s).





The following concepts are illustrated above:

**A payment event defines the event**

A payment event is required whenever any form of payment is received. The payment event defines the payment date and effective date (and that's all).

**A payment event has tender(s)**

A tender exists for every form of tender remitted as part of the payment event. A payment event must have at least one tender otherwise nothing was remitted. A payment event may have many tenders when multiple payment methods are associated with an event (e.g., paying with cash, a check, and a credit card).

**A payment is allocated to account(s)**

The total amount of tenders under a payment event is distributed to one or more accounts.

**A payment is distributed to obligations**

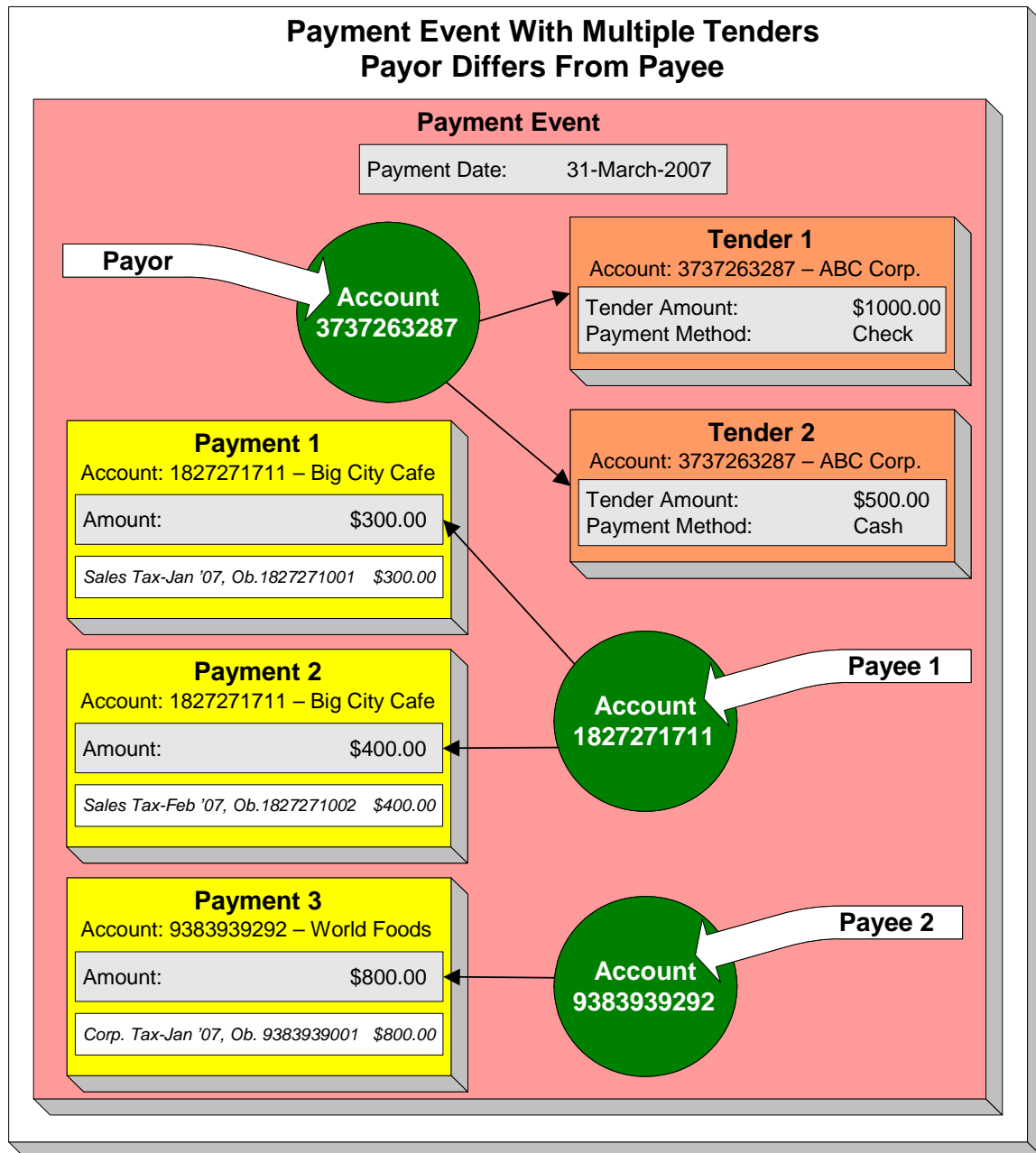
The system allocates an account's payment amount amongst its obligations. The system creates a payment segment for each obligation that receives a portion of the payment.

**Payor and payee are frequently the same**

The account remitting the tender (the payor) is frequently the same as the account to which the funds are allocated (the payee). The next illustration provides an example when this is not the case.

**Multiple Tenders Used To Pay For Multiple Accounts**

The following diagram illustrates a payment event with multiple tenders where the payor of the tender is not the same as the account(s) receiving the payment.

**A payment event may have many tenders**

A single payment event may have many tenders. While the above example shows both tenders being paid by the same account, each tender may reference a different account.

**Many accounts may be paid under 1 event**

The total amount of tenders under a payment event are distributed to one or more accounts.

**Payor may differ from payee**

The account(s) remitting the tender may differ from the account(s) whose debt is relieved.

## An Overview Of The Payment Event Creation & Allocation Process

When a payment event occurs, the system stores a tender for each form of remittance (e.g., cash, check, charge). It then allocates the sum of the tenders to one or more accounts.

By default, the system allocates the sum of the tenders to the account that remits the tenders. You may override this default and specify any number of accounts and their respective payment allocation amount. This is useful, for example, when a social service agency pays for many accounts. If applicable, you may also configure the system to use your own payment event distribution rule(s).

Refer to [Distributing A Payment Event](#) for more information.

The system distributes a payment amongst an account's obligations based on the age of each obligation's debt AND distribution priority. The system creates a payment segment for each obligation that receives part of the payment.

Refer to [Distributing A Payment Amongst An Account's Obligations](#) for more information.

You may manually redistribute the payment amount amongst the account's obligations before you commit the distribution. When the distribution is acceptable, you freeze the payment. Freezing a payment causes the system to create a financial transaction for each related payment segment. It is the financial transaction(s) that causes the obligations' payoff and current balances to be reduced. The financial transaction also contains the journal details that debit "cash" and credit some other GL account.

And that's it. The remaining topics in this section provide more information about the creation and allocation of payment events.

**Batch and real-time payment event creation / allocation.** There is only one payment event creation / allocation routine and therefore anything the batch payment process does for whole batches of payments, you can do to a payment on-line.

## Payments and Penalty and Interest

If your organizations levy penalty and interest (P&I) charges for unpaid tax assessments, creating or canceling payments should recalculate penalty and interest.

The recalculation occurs when a payment is frozen or canceled assuming that the correct algorithm has been plugged in to the account's [account type](#).

## Payment Date and Effective Date for Payment Events

This section discusses the payment and effective dates used in payment events.

By default, the payment and effective dates are equal but the algorithm used to create the payment may calculate the dates differently based on business rules.

A payment event records a payment date and an effective date. The payment date should be the date that the payment was considered "received". This could be the system date or it could be a postmark date. The effective date is used to populate the effective date of the payment segment's financial transactions and is the date that the payment should affect penalty and interest.

The following examples illustrate when the effective date may differ from the payment date:

- Consider the following collections scenario. A collections notice is sent to a taxpayer with penalty and interest forecasted to a future date (for example, the 30th). If the payment is received (postmarked) on the 20th, this date is used as the payment date. However, your organization's business rules may dictate that for P&I purposes, the payment should be considered effective on the 30th (the date noted in the collection notice). In this case the distribution algorithm that directs this payment to a collection notice should set the payment effective date to the 30th. This ensures that penalty and interest will be calculated according to what was forecast.
- Consider an obligation whose filing period due date is April 15th, where the payment grace days on the obligation type is 3. If the taxpayer files and pays on April 17th, the payment distribution algorithm determines that this is within the grace days and sets the effective date to the 15th. In this case the payment date would still be the 17th.

**Changing effective date.** A BPA script [\*Change Payment Effective Date\*](#) is provided as an option on the payment event context menu. The script prompts you for a new effective date. It updates the payment event and its related FT and then causes penalty and interest to be recalculated.

## Distributing A Payment Event

**Warning!** This section deals with the concept distributing a payment amount into payment(s). It does not discuss the distribution of a single payment into segments. **For more information** about payment distribution, refer to [Distributing A Payment Amongst An Account's Obligations](#).

The base-package, by default, creates a single payment for a payment event. Some business practices require potentially many payments to be created when payment events are added.

A few examples of when multiple payments may be necessary are:

- A payment amount needs to be distributed towards different distribution types:
  - \$50 in interest
  - \$60 in collection charges
  - \$70 in taxes

Each of the above distributions is realized as a separate payment identified by its own match type (i.e., there'll be one match type called Interest, another called Collection Charges, etc.).

In this example, the debt of a single obligation may be relieved by each of these payments.

- In a similar way, you may want to create a separate payment for an overpayment to differentiate it from regular payments (using yet another match type).
- In the case of a social service agency that pays for many accounts, a single payment event may be distributed amongst multiple accounts.

The method by which a payment amount is distributed to create payment(s) is contained in [Create Payment algorithms](#) plugged in on a distribution rule.

There is yet another aspect to having control over how payment events are created. The default method of creating payment events assumes knowledge of account IDs (of the payor and the payee) when making a payment. In cases where payments are made by and towards business entities other than accounts, knowledge of their corresponding account IDs may not be available at payment time. Consider the following examples:

- A payment is made for a taxpayer. The payment is applied to all of the taxpayer's accounts.
- A payment is made for a specific filing period.
- A court-ordered payment is made for a case. The payment is directed toward the assessments linked to the case, where assessments may cross tax roles, filing periods, etc
- A payment is made for a collections notice. The payment is directed toward specific assessments linked to the notice.
- A payment is made for a pay plan. The payment is applied to the specific obligations that are covered by the pay plan.

The method by which the tender account is determined by means of an alternate identifier is contained in [Determine Tender Account algorithm](#) plugged in on a distribution rule.

Refer to [Making Payments Using Distribution Rules](#) for information on how to configure your system to use this distribution method.

## Distributing A Payment Amongst An Account's Obligations

**Warning!** This section deals with the concept of distributing a payment amongst an account's obligations. It does not discuss how the sum of a payment event's tenders is balanced out by payment allocations. **For more information** about payment event balance, refer to [Unbalanced Payment Events](#).

A payment must be distributed to one or more obligations for its financial impact to be realized. When a payment satisfies an account's entire debt, you don't have to worry about how the system distributes the payment. The concept of payment distribution is only relevant when a partial or excess payment is distributed.

The first important point to understand is that the method of distributing a payment amongst an account's obligations is contained in an algorithm that's plugged in on to [account type](#). This means that you can have different distribution algorithms for different account types.

**Manual overrides.** Most of the time, you'll let the payment distribution algorithm distribute the payment amongst an account's obligations. However, you may manually distribute a payment when a taxpayer directs a payment to specific obligation(s).

The following explanation describes one of the base package payment distribution algorithms (refer to [Payment Distribution – Pay Priority and Debt Age](#) for information about this algorithm). This algorithm distributes a payment based on:

- The age of each obligation's debt.
- The payment distribution priority of each obligation's obligation type.

The following diagram helps illustrate how the distribution algorithm works.

**Important!** There are other payment distribution algorithms in the base package. Click [here](#) to see the available algorithm types.

**Distributing A Payment Amongst An Account's Obligations  
Is Controlled By The Age Of The Debt And The Obligations's Distribution Priority**

		<b>Priority 10 (Highest)</b>		<b>Priority 90 (Lowest)</b>	
		<b>Obligation #1</b>	<b>Obligation #2</b>	<b>Obligation #3</b>	
Excess Credits		<b>16</b>			
New Debits*		<b>13</b>	<b>14</b>	<b>15</b>	
Non-Delinquent		<b>10</b>	<b>11</b>	<b>12</b>	
A r r e a r s	35-day Arrears	<b>5</b>	<b>6</b>	<b>9</b>	35-day Arrears
	66-day Arrears	<b>3</b>	<b>4</b>	<b>8</b>	52-day Arrears
	93-day Arrears	<b>1</b>	<b>2</b>	<b>7</b>	93-day Arrears

\*--Includes all charges not yet invoiced to the taxpayer.

The above example shows three columns, one for each obligation linked to a hypothetical account. Notice that two of the obligations have the same distribution priority, the third has a lower priority. The numbers in the cells indicate the order in which the system distributes a partial payment.

**Debt terminology.** Before we can discuss the distribution algorithm, you must understand the terminology we use to categorize debt. **Delinquent** debt is associated with financial transactions that appear on overdue bills. **Non-Delinquent** debt is associated with financial transactions that appear on current bills. **New Debits** debt is associated with financial transactions that do not yet appear on a completed bill.



The following points describe the algorithm used to distribute the partial payment:

- The system pays off delinquent debt of the highest priority obligations first. In the above example, where multiple obligations have the same distribution priority, the system does NOT payoff one obligation before it starts on the next (which one would it pick?). Rather, it distributes the payment amongst the obligations based on the age of the respective debt on each obligation. In the above example, this is represented by steps 1 through 6 (notice how the distribution jumps between Obligation #1 and Obligation #2).
- After all delinquent debt has been relieved from the highest priority obligation(s), the system pays off the next priority until all delinquent debt is relieved. In the above example, this is represented by steps 7 through 9.
- The system next pays off non-delinquent debt using each obligation's respective distribution priority. Note well, the payment distribution algorithm doesn't associate an age with non-delinquent debt and therefore the distribution is based purely on the obligations' respective distribution priority. In the above example, this is represented by steps 10 through 12.
- After all non-delinquent debt is relieved, the system next pays off "new debit" debt based on the obligations' respective distribution priority. In the above example, this is represented by steps 13 through 15.
- Refer to [Overpayment](#) for a description of what happens if money still exists after the above distribution is complete.

**Payment segments and financial transactions.** A payment segment exists for each obligation that receives a portion of a payment. Linked to each payment segment is a financial transaction. It is the financial transaction that causes the obligation's debt to be relieved and the general ledger to be impacted.

Refer to [Payment Exception](#) for more information about how the system handles errors detected during the payment distribution process.

**Overriding the distribution algorithm for uploaded payments.** The standard distribution algorithm is used for payments that are [Interfaced From An External System](#) unless you specify a MATCH\_VALUE and MATCH\_FLG on the [Payment Staging](#) row associated with the uploaded payment. These fields are used in conjunction to indicate that the distribution of the payment should be restricted in some way (i.e., the standard payment distribution algorithm should not be used). MATCH\_FLG indicates how the payment should be distributed (e.g., only distribute to a specific obligation), MATCH\_VALUE contains the ID of the restriction (e.g., the obligation ID).

**Open item customers.** For an [open-item taxpayer](#), you MUST override the standard distribution algorithm because the payment is distributed as per the open items that it is relieving. Refer to [Payments And Match Events](#) for more information.

## Overpayment

Overpayment refers to the situation where money is left over after a payment has been distributed to all eligible obligations, and all debt is relieved. Refer to [Overpayment Obligations](#) for a description on how to configure the system to handle your overpayment requirements.

## Canceling A Tender Versus Canceling A Payment

A payment event has tender(s) and payment(s). You can cancel a tender when it's not valid, e.g., when a check bounces. You can cancel a payment when the account should not have received the payment (e.g., a misdistribution or a canceled tender).

When you cancel a tender, the system automatically cancels ALL **frozen** payments. We do this because if the tender is canceled, there are no funds to distribute to accounts (unless there are other non-canceled tenders under the event). However, when you cancel a payment, the system does NOT cancel the tender(s) because we assume that, if the tenders were incorrect, you would have canceled them rather than the payment.

## NSF Cancellations

When a tender is canceled, a cancellation reason must be supplied. If the cancellation reason indicates a NSF (non sufficient funds) charge should be levied, the system invokes the NSF charge algorithm specified on the tender's account's [account type](#). Algorithms of this type will typically create an adjustment or billable charge to levy the NSF charge.

Besides calling this algorithm, an NSF cancellation may affect the tendering [account's compliance rating](#). The cancellation reason indicates the extent to which the account's ratings are affected.

## Transferring A Payment

If the account on an event's tender and payment are wrong, you can use the [Transfer](#) button on the Payment Event page to transfer the payment to another account.

If the account on the payment is wrong – but the tender is correct, you can use the [Transfer](#) button on the Payment page to transfer the payment only to another account.

## Unbalanced Payment Events

The system, by default, distributes the sum of a payment event's tenders to the account that remits the tender with a single payment. After distribution the sum of the tenders equals the sum of the payments when the event is first created. We refer to such an event as being **balanced**.

However, it is possible for an event's tender amount to not equal the sum of the payment allocations (i.e., the event becomes **unbalanced**). How? Well, there are several ways this can happen:

- While the system DEFAULTS the payment amount to be the tender amount, you can override the payment amount and therefore make a previously balanced event **unbalanced**.
- While the system DEFAULTS the payment account to be the tender account, you can add additional accounts / amounts and therefore make a previously balanced event **unbalanced**.

- When you cancel a tender (e.g., because a check bounces), the system cancels ALL payments linked to the tender's payment event. If the payment event has multiple tenders, this will cause the event to become **unbalanced**. To correct this situation, you must add payment allocations to equal the amount of non-canceled tenders.
- If you cancel a payment and forget to add another payment for the same amount, the event becomes **unbalanced**. To correct this situation, you must add another payment (or cancel the tender).
- You may delete a tender from an event while its tender control is **open**. If you delete a tender and don't do anything about the related payments, the event becomes **unbalanced**.
- You may add a tender to an event at any time. If you don't allocate the tender amount to an account, the event becomes **unbalanced**.

Refer to [Payment Event Exceptions](#) for more information about how the system reminds you about unbalanced payment events.

## Tender Management and Workstation Cashiering

When you add a tender, you must identify its **Tender Source**. For example,

- A specific cash drawer ID is the source of tenders remitted to a cashier.
- The notional lockbox ID is the source of tenders interfaced from a lockbox.
- The remittance processor is the source of tenders interfaced from a remittance processor.

For more information, refer to [Setting Up Tender Sources](#).

A tender source's tenders must be balanced against an expected total before they can be deposited at a bank. This periodic balancing requires all tenders to exist in respect of a **Tender Control**. Over time, a tender source may have many tender controls (one per balancing event).

An example of a cashier's cash drawer will help clarify the tender control concept:

- When a cashier starts in the morning, s/he starts with a fresh cash drawer (i.e., one without tenders). Whenever a drawer starts afresh, a new Tender Control must be created because you balance the contents of a drawer.
- A cash drawer typically contains funds to make change. These funds are the tender control's **Opening Balance**.

**Default note.** A tender control's starting balance defaults from its tender source.

- During the day, taxpayers remit tenders to the cashier. Every tender put into the drawer is associated with the drawer's tender control created at the start of the day.
- A tender control's balance increases during the day as tenders are recorded. A cashier can view the balance at any time.

- The cashier can turn in funds to the head cashier during the day. Each turn in event can be recorded in the system. Note well, if the amount of funds in a tender control exceeds the maximum balance defined on the tender control's tender source, a warning is issued to the cashier to remind him/her to turn in funds.
- At some point, the contents of the drawer must be balanced against the total tenders linked to the tender control. When balancing starts, no additional tenders may be put into the tender control. If the cashier receives additional tenders after balancing starts, a new tender control must be created (and the above process starts afresh).
- During the balancing process, some modifications may be made to the tenders associated with the tender control, but no additional tenders may be added. When the tender control is balanced, neither it nor its tenders may be modified.

For more information, refer to [Managing Your Cash Drawers](#).

While the above explanation is true, it isn't complete. In addition to the requirement that a tender must reference a tender control, the tender control must refer to a **Deposit Control**. Deposit controls give you administrative control over all of the tender controls whose contents will be deposited en masse. The following concepts will help explain the power of deposit controls:

- As explained above, when a cash drawer is started afresh, a new tender control must be created. The tender control "holds" all new tenders received by the cashier.
- Similarly, when a tender control is created, it must reference a deposit control. During the day, a deposit control's tender controls are constantly changing. You can view the total impact of a deposit control's tender controls at any time.
- At some point, you will want to deposit the tenders received during the day. To do this, you must indicate how much will be deposited at the bank. This deposit amount must equal the sum of the tender controls linked to the cash drawer. When this deposit balancing starts, no additional tender controls may be associated with the deposit control.
- When the deposit total equals the sum of the tender controls, the deposit control becomes **balanced** and no changes may be made to it, its tender controls, or its tender controls' tenders.

For more information, refer to [Managing Your Cash Drawers](#).

**Background processes use the same concepts.** Tenders that are interfaced from external sources (e.g., lockboxes and remittance processors) make use of the concepts describe above. For example, tenders interfaced from a remittance processor are linked to a tender control and this tender control is linked to a deposit control. The main difference is that the background processes require no human intervention; the system automatically creates tender and deposit controls and sets their states to **balanced** when the interface concludes successfully.

**The ACH activation process also creates tender and deposit controls.** Refer to [Activating Automatic Payments](#) for more information.

The topics in this section elaborate on the tender management concepts described above.

**Contents**

Managing Your Cash Drawers  
Turn Ins  
Balancing By Tender Type  
Cash Back  
Managing Payments Interfaced From External Sources

**Managing Your Cash Drawers**

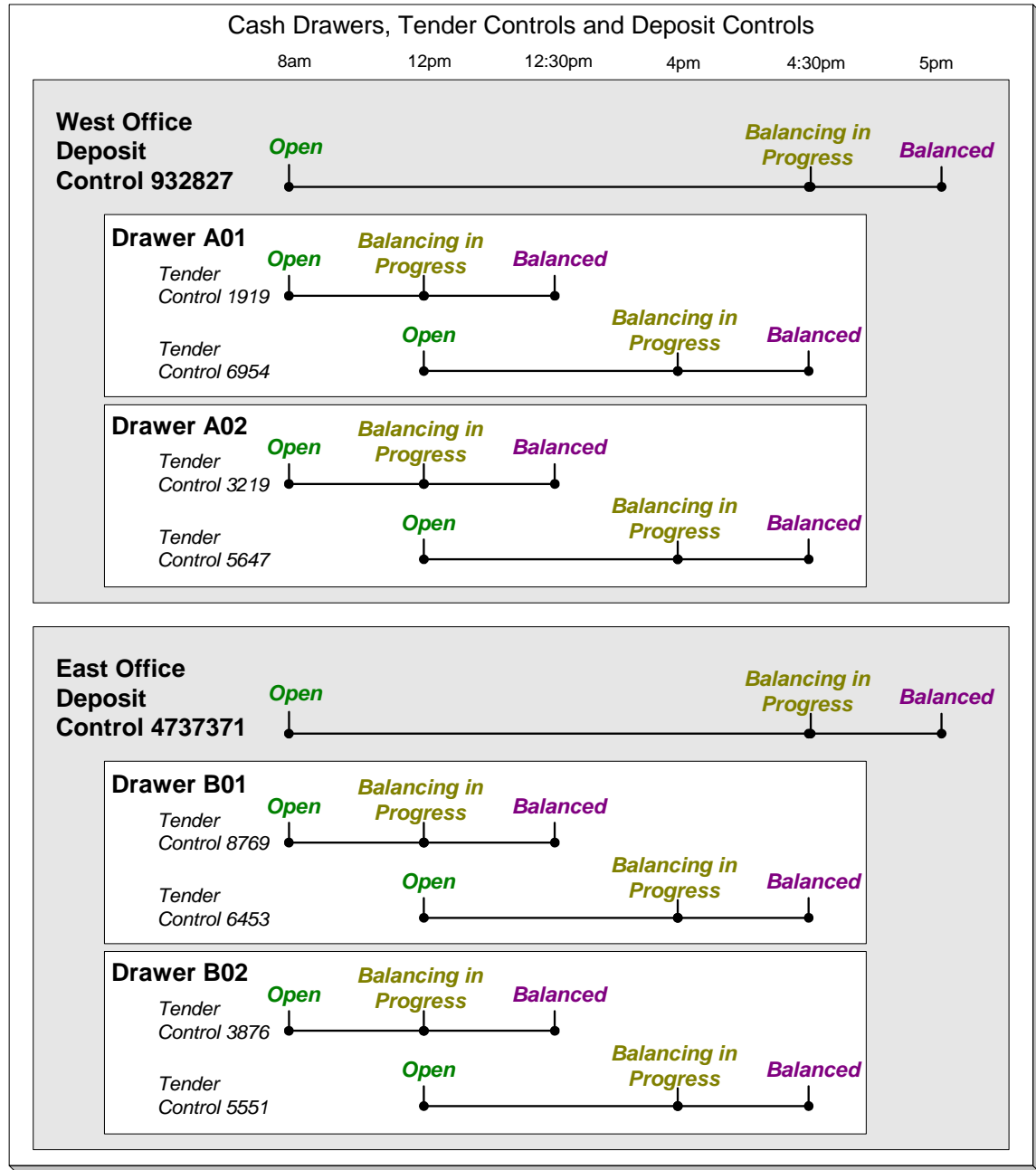
**Warning!** This section assumes you are familiar with the concepts described in [The Lifecycle Of A Deposit Control](#) and [The Lifecycle Of A Tender Control](#).

There are many ways to handle the daily management of tenders received via cash drawers. It really depends on how your organization works. To help you understand the potential of the system, we'll continue the example started above.

Assume that the cash drawers in your western office are balanced and deposited independently from those in your eastern office. We'll assume that both offices follow the same daily routine:

- Load fresh drawers first thing in the morning. Each drawer contains a starting balance of \$150.00. Note: the drawer's tender control's starting balance defaults from its tender source.
- At 10 am, the cashier turns in funds to the chief cashier and continues to receive additional tenders.
- At 12 noon, each drawer is pulled and balanced by a supervisor.
- By 12:30 pm, the tender controls are balanced.
- At 4 pm, the cashiering stations are closed. Each drawer is pulled and balanced by a supervisor.
- By 4:30 pm, the tender controls are balanced.
- At 5 pm, the deposit control is balanced and funds are ready to be deposited at the bank.

Given this, the following diagram illustrates the deposit controls and tender controls used by each office on a given day.



The following concepts are illustrated above:

- An **Open** deposit control must exist before you can create a tender control. And an **Open** tender control must exist before you can create a tender. From a business process standpoint, this means:
  - A supervisor would create a deposit control at the start of the day (8 am in the above illustration).
  - Each cashier would create a tender control when they start a drawer and reference the deposit control created by the supervisor.

- During the day, the cashier can turn-in moneys to the chief cashier. These turn-in events are recorded in the system as they play a part in the ultimate balancing of the drawer. Refer to [Turn Ins](#) for more information.
- At some point, the contents of a drawer can be pulled and balanced. If additional tenders can be received in a drawer, a new tender control must be created for the drawer. Refer to [Balancing By Tender Type](#) for more information.
- At the end of the day, the supervisor checks to make certain that all tender controls linked to the deposit control are **Balanced**. After this has been done, the supervisor indicates the deposit amount on the deposit control and changes it to **Balanced**. Notice that in the above illustration each deposit control references four tender controls.
- Typically, a cash drawer has one tender control **Open** at any point in time (meaning that the tenders being received are being linked to a specific tender control). However, this is not a hard rule. If you want, you may have multiple tender controls **Open** at any point for a specific cash drawer (for example, if multiple cashiers can work the same drawer during the day but take their drawer with them).
- Typically, a specific cashier puts tenders into a specific tender control. However, this is not a hard rule. On a tender control, you can define if it's limited to a specific operator OR if any operator can link tenders to it.
- When you're ready to balance a drawer, you change the tender control to **Balancing in Progress**. This prevents new tenders from being added to the tender control. If the cashier can continue to receive tenders, s/he must create another tender control. In the above example, all drawers are balanced at 12 noon by a supervisor while the cashier continues to take payments.
- When the tender control is balanced, you change its state to **Balanced**. This prevents any changes to the tender control or its tenders.
- All tender controls exist in respect of a deposit control (in fact, the deposit control must be created before the tender control). This way, a supervisor can check the state of the related drawers throughout the day. Notice that the state transition of a deposit control is identical to that of the tender control (refer to [The Lifecycle Of A Deposit Control](#) and [The Lifecycle Of A Tender Control](#)). There is only a temporal difference. Notice that the deposit control stays open throughout the day while any number of tender controls are being opened and balanced.

**Multiple deposits in a day.** While the above example illustrates a single deposit per office per day, it is quite possible to have multiple deposit controls on any given day.

**Turns ins.** The above example did not illustrate the fact that a cashier can turn-in moneys during the day without having to balance the drawer. Refer to [Turn Ins](#) for more information.

## Turn Ins

A cashier may optionally turn-in funds received into a cash drawer to a head cashier. The turn-in process requires two steps:

- Each time a cashier turns-in funds, they add a turn-in event on the [Tender Control – Turn Ins](#) page. Note that a separate turn-in event is required for each type of tender that's turned in. This is because the balancing of a tender control is performed for each tender type and therefore the system must know how much of each tender type has been removed from the drawer.

**Turn in warning.** If the amount of cash-like funds in a tender control exceeds the maximum balance defined on the tender control's tender source, a warning is issued to the cashier to remind him/her to turn in funds.

- The head cashier (the person responsible for the deposit control associated with the various tender controls) approves the turn in using the [Deposit Control – Turn Ins](#) page.

**All turn-ins must be approved for balancing to complete.** A tender control cannot be balanced until the head cashier has approved all turn-in events.

## Balancing By Tender Type

It should be noted that when it's time to balance a tender control, the cashier must enter the amount of each tender type that is in the drawer in order to balance it. For example, assume the following takes place:

- A drawer is opened with a starting balance of \$150.50 (tender type is Cash).
- During the day, the cashier receives \$5,000 in Cash, and \$1,000 in Checks.
- During the day, the cashier turns in \$750 of the Checks and \$4,000 of Cash.

At the end of the day, the following operator would have to enter the balances shown in the Ending Balance column.

Tender Type	Starting Balance	Tenders Received	Turn Ins	Ending Balance
Cash	\$150.50	\$5,000	\$4,000	\$1150.50
Check	-	\$1,000	\$750	\$250

**Time saver.** The system assists in the balancing effort by amalgamating the amount of tenders by tender type when the tender control's status is changed from **Open** to **Balancing In Progress**. The cashier just needs to enter the Ending Balance. The system can then compare the cashier's Ending Balance against the Expected Ending Balance. When these values are equal for all tender types, the tender control's status can become **Balanced**.

## Cash Back

When a [payment is added](#), the user defines the following:

- The amount of debt to be relieved (i.e., the payment amount)
- The amount remitted (i.e., the tender amount)
- The form of the remittance (i.e., the [tender type](#))



The payment amount typically equals the tender amount unless cash will be returned to the taxpayer. For example, if a taxpayer remits \$100, but only wants to pay off \$25 of debt, the tender amount will be \$100 and the payment amount will be \$25. The system will only allow a user to remit more than the payment amount if the tender type indicates "cash back" is allowed. For example, you may not allow cash to be returned if a check is remitted, but you may allow it to be returned if cash is remitted.

If cash back is allowed for the tender type, the system displays the amount of cash to be returned on the [Payment Event](#). In addition, because the system enforces [balancing the cash drawer by tender type](#), the system adjusts the payment event's tendered information as follows:

- When there is cash back, the payment event will have two tenders - one will be for the amount and type entered by the user, the other will be a negative amount with a tender type of cash (note, this tender type is retrieved from the Starting Balance Tender Type on the [installation record](#)). For example, if a taxpayer remits \$100 in traveler's checks, but only wants to pay off \$25 of debt, there will be two tenders: one for the \$100 travelers check and the other for the -\$25 of cash.

**Multiple tenders and payment cancellation.** If multiple tenders were created because of "cash back" processing, both tenders must be cancelled if the payment event needs to be cancelled.

When modifying an unfrozen payment on the [Payment Event](#), if the payment becomes unbalanced, a button is displayed allowing the user to **Recalculate Cash Back**. If the user clicks this button, the system reassesses the cash back tender as follows:

- If there is now cash back, a new tender is created for the credit amount
- If the cash back amount has changed, the tender for the cash back is adjusted to the new amount
- If there is no longer cash back due, the tender for the cash back is removed.

## Managing Payments Interfaced From External Sources

Just like a payment recorded on-line via a cash drawer, a payment interfaced from an external source (e.g., lock box or remittance processor) must reference a tender control and the tender controls, in turn, must reference a deposit control. The only real differences between these two types of payments are highlighted below:

- Whilst an operator must create and balance the tender and deposit controls for real-time payments, the system creates the tender and deposit controls associated with interfaced payments.
- It's impossible for an invalid account to be referenced on a payment recorded real-time. However, it is quite possible for an interfaced payment to reference an unknown account. If an invalid account is referenced on an interfaced payment (using no distribution rules), the system links it to the suspense obligation referenced on the tender source control table.

Refer to [Interfacing Payments From External Sources](#) for more information.

## Exceptions

The topics in this section describe exceptions that are detected when the system allocates a payment.

### Contents

- [Payment Exceptions](#)
- [Payment Event Exceptions](#)
- [Resolving Exceptions Automatically](#)

### Payment Exceptions

When the system attempts to distribute a payment, there are a small number of situations where it can't do its job. Some examples of classic errors:

- **No obligation to hold a credit.** For example, if an account overpays their debt and the account doesn't have a single obligation that is allowed to hold a credit, a payment error is generated.

The system saves payments that are in error just as it saves payments that are error-free. This is done because payments are nothing more than a snapshot of the data that was used to distribute the payment. By saving the snapshot, you can see the information the system used when it detected the error and therefore more effectively correct it.

Every payment in error is written to the [Payment Exception](#) table. A To Do background process creates To Do entries for records in this table.

### Payment Event Exceptions

It is possible for a payment event's tenders to not equal its payments. Such events are classified as **unbalanced**. Refer to [Unbalanced Payment Events](#) for how this can happen.

For each unbalanced payment event, a record is written to the [Payment Event Exception](#) table. A To Do background process creates To Do entries for records in this table.

### Resolving Exceptions Automatically

Some payment errors occur because master data was not fully set up prior to receiving a payment for the account. For these cases, the system will periodically check to see whether the master data problem has been resolved by attempting to distribute and freeze the payment in error.

A background process, Resolve Payments in Error – [PY-RPE](#), exists for this purpose. This background process works as follows:

- It looks for payments in error where the error was caused by the lack of active obligations linked to the payment's account.
- For each such payment, it attempts to re-distribute the payment. If obligations have been created in the meantime, this payment will distribute and freeze successfully.

## Payment Financial Transaction Considerations

A payment segment exists for each obligation that receives a portion of a payment. Linked to every frozen payment segment is a financial transaction. This financial transaction affects an obligation's payoff balance and/or current balance. It also contains the journal details that debit cash and credit some other account.

The topics in this section provide information about the financial impact of a payment segment.

Refer to [The Financial Big Picture](#) for more information about a payment's place in the financial big picture.

### Contents

[Payment - Current Balance versus Payoff Balance](#)

[The Source Of GL Accounts On A Payment Financial Transaction](#)

## Payment - Current Balance versus Payoff Balance

**Warning!** If you do not understand the difference between payoff balance and current balance, refer to [Current Amount versus Payoff Amount](#).

A payment segment financial transaction almost always affects payoff balance and current balance by the same amount (think of it like this - when a taxpayer pays, the amount they think they owe goes down by the amount they really owe). The only exception is a payment segment for a charitable contribution. These payment segments only affect current balance because the taxpayer was never assessed for the contribution in the first place.

Refer to [Setting Up Payment Segment Types](#) for more information about how payment segment type affects how a payment segment is produced and how its financial transaction is generated.

## The Source Of GL Accounts On A Payment Financial Transaction

A payment segment's financial transaction also contains the double-sided accounting entry that defines how the payment segment affects the general ledger.

Refer to [The Source Of GL Accounts On Financial Transactions](#) for a description of where the system extracts the distribution codes used to construct the GL accounts.

## A Payment May Affect More Than Just Taxpayer Balances

The topics in this section provide information about obscure things that may happen when a payment is distributed and frozen.

### Contents

[Open Item Accounting and Match Events](#)

[FT Freeze Repercussions](#)

## Open Item Accounting and Match Events

Refer to [Payments and Match Events](#) for more information about how payments can create match events for open-item accounts.

## FT Freeze Repercussions

Refer to [Obscure Things That Can Happen](#) for more information about things that can happen when an FT is frozen (and FT's get frozen when a payment is frozen).

## Automatic Payments

This section discusses how to set up and manage taxpayers who pay their bills automatically (via direct debit or credit card debits)

### Contents

- [How To Set Up A Taxpayer To Pay Automatically](#)
- [What Are Automatic Payments?](#)
- [How And When Are Automatic Payments Created?](#)
- [Automatic Payment Dates](#)
- [How To Implement Maximum Withdrawal Limits](#)
- [How Are Automatic Payments Cancelled?](#)
- [Match Events Are Created For Open-Item Customers When An Automatic Payment Is Created](#)
- [Pay Plans and Automatic Payment](#)
- [Downloading Automatic Payments and Interfacing Them To The GL](#)
- [ACH Record Layouts](#)

### How To Set Up A Taxpayer To Pay Automatically

If a taxpayer wants to pay automatically, transfer to [Account – Auto Pay](#) and define the source of the funds and the taxpayer's account or credit card number.

### What Are Automatic Payments?

An automatic payment is just like any other payment (refer to [A Payment Event Has Payments And Tenders](#) for more information about payments in general). However, automatic payments have one special trait – they cause funds to be transferred into your company's bank account. Refer to [Downloading Automatic Payments and Interfacing Them To The GL](#) for how this transference happens.

### How And When Are Automatic Payments Created?

Automatic payments can be created in several ways:

- The system creates automatic payments for bills linked to accounts with an active auto pay option.
  - At bill completion time, the bill is stamped with the automatic payment's extract date and amount. The date is the automatic payment source's extraction date (refer to [Automatic Payment Dates](#) for more information on how this date is calculated).
  - The automatic payment background process ([APAYCRET](#)) creates the automatic payment on the extract date stamped on the bill.

**An algorithm is used to create automatic payments.** The logic used to create automatic payments is plugged in on the [Installation Record](#). The base package includes two sample algorithms. [APAY-CREATE](#) creates each auto pay as one payment event with one tender and one payment. Use [APAY-CREATE](#) if you use standard payment distribution. [C1-APAY-CRDR](#) creates automatic payments using distribution rules. Use [C1-APAY-CRDR](#) if you use distribution rules.

- The automatic payment is NOT distributed and frozen when the automatic payment is initially created. A separate background process ([APAYDSFR](#) / [C1-APYDF](#)) distributes and freezes the automatic payment on the automatic payment GL distribution date (refer to [Automatic Payment Dates](#) for more information on how this date is calculated). This means that the taxpayer's balance increases when the bill is completed and is only reduced when the automatic payment is marked for interface to the general ledger.

**APAYDSFR uses standard payment distribution.** This process assumes the automatic payment was created as: one payment event, one payment. Use this to distribute / freeze payments created by [APAY-CREATE](#).

**C1-APYDF uses distribution rules.** This process assumes that the automatic payment was created using distribution rules. Use this to distribute / freeze payments created by [C1-APAY-CRDR](#).

- Note that it is possible for automatic payments to be distributed and frozen after being extracted and interfaced to a financial institution. Please refer to [Downloading Automatic Payments and Interfacing Them To The GL](#) and [The Nightly Processes](#).

An algorithm plugged in on the [Installation Record](#) calculates the payment amount whether the automatic payment is created at bill completion time or on the extract date. Please refer to [APAM-DFLT](#) for more information about how the algorithm that is supplied with the base package calculates this amount.

**With balance forward accounting, automatic payments are not just for new charges.** The base package algorithm includes prior balances when it creates a taxpayer's first automatic payment. For example, if a taxpayer has an existing balance of \$100 and then signs up for automatic payment, their next bill will cause an automatic payment of \$100 plus any new charges to be created (assuming the \$100 remains unpaid at the time the next bill is completed). Refer to [Open Item Versus Balance Forward Accounting](#) for information about balance forward accounting.

- If a taxpayer with an account that is set up for automatic payment has a pay plan that is not excluded from automatic payment, a background process ([C1-PAYPA](#)) creates an automatic payment on the scheduled payment dates. Please note, the automatic payment is NOT distributed and frozen when the automatic payment is initially created. Rather, a separate background process ([APAYDSFR](#) / [C1-APYDF](#)) distributes and freezes the automatic payment on the automatic payment GL distribution date (refer to [Automatic Payment Dates](#) for more information on how this date is calculated). Refer to [The Big Picture of Pay Plans](#) for more information about pay plans.

- A user can create an automatic payment by simply adding a payment tender with a tender type that indicates it is for automatic payment purposes. This would be a rather unusual thing to do, but you might do this if you want to immediately debit a taxpayer's bank account after a large adjustment is added to the system (e.g., if they suddenly owe you a lot of money and you don't want to wait until the next bill to collect it). Automatic payments created by this method must be distributed and frozen before they can be extracted.

**Auto pay creation algorithm is not invoked for manually created automatic payments.**

Please note that this algorithm is **not** called when a user manually creates an automatic payment (by adding a payment tender with a tender type that indicates that it is for automatic payment purposes).

When an automatic payment is first created, it gets marked with a distribution date. The distribution date is the date on which the automatic payment's FT's GL details can be interfaced to the general ledger (via the standard GL interface). The distribution date is determined as follows:

- Every automatic payment references an auto-pay source.
- Every auto-pay source references an auto-pay route type.
- Every auto-pay route type contains an algorithm that is responsible for calculating the GL Distribution (Posting) date. On the GL distribution date, the automatic payment will be interfaced to the general ledger.

## Automatic Payment Dates

As described in the previous section, an algorithm (that's plugged in on [Auto Pay Route Types](#)) controls the date on which the automatic payment is interfaced to your general ledger. We refer to this date as the GL distribution date.

This algorithm also populates the following dates:

- The payment date that is stored on the payment.
- The date on which the automatic payment is interfaced to the financial institution.

The algorithm that is supplied with the base package provides many parameters that allow you to dictate how these dates are calculated. Please refer to [APAY-DTCALC](#) for the details.

## How To Implement Maximum Withdrawal Limits

In some locales, taxpayers can define a "maximum withdrawal amount" to limit the amount of money that is automatically debited from their bank account. For example, a low-income taxpayer may want to prevent direct debits of more than \$50 from being applied to their checking account.

You define a taxpayer's maximum withdrawal amount when you setup their automatic payment information on [Account – Auto Pay](#).

The following points describe how the system implements maximum withdrawal limits:

- When a bill is completed for a taxpayer who pays automatically the system calls the automatic payment creation algorithm that's plugged in on the [Installation Record](#). The [base-package autopay creation algorithm](#) checks if the amount of the automatic payment exceeds the taxpayer's maximum withdrawal amount. If so, the autopay creation algorithm calls the [automatic payment over limit algorithm](#) that's plugged in on the account's [account type](#). Algorithms of this type have the ability to reduce the amount of the autopay or to prevent the autopay from being created. Refer to [APOL-RA](#) for an example plug-in.
- When a user manually creates an automatic payment (by adding a tender with a tender type that indicates that it is for automatic payment purposes), the system issues a warning message when the tender amount exceeds the account's maximum withdrawal amount.

**Pay plans.** Please note that automatic payments that are created as a result of pay plans are not subject to maximum withdrawal limits. This is because both of these options required taxpayer approval and therefore the taxpayer should be able to plan accordingly. Refer to [How And When Are Automatic Payments Created](#) for more information.

## How Are Automatic Payments Cancelled?

There are two ways to cancel an automatic payment:

- The system will cancel an automatic payment behind-the-scenes if the related bill (if any) is reopened BEFORE the automatic payment is interfaced to the financial institution. When you recomplete the bill, the system will create a new automatic payment that reflects the new amount due (and the canceled automatic payment will net out the original automatic payment).

An operator can cancel an automatic payment (refer to [How To Cancel A Tender](#)) at any time. You would do this if the financial institution rejected the automatic payment.

## Match Events Are Created For Open-Item Customers When An Automatic Payment Is Created

The system creates a match event when a bill is completed for open-item customers that pay automatically (i.e., direct debit taxpayers). The match event groups together the bill's new charges against the automatic payment's payment segments.

If the bill is subsequently re-opened, the match event will be cancelled when the automatic payment is cancelled.

Refer to [Open Item Accounting](#) for more information.

## Pay Plans and Automatic Payment

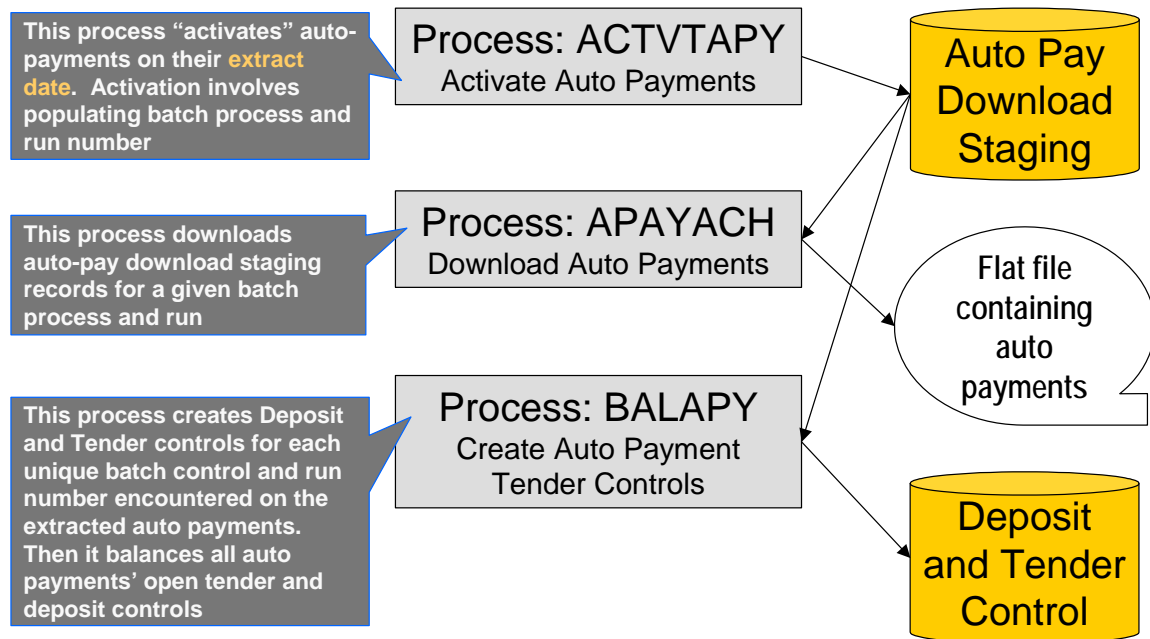
If a taxpayer wants to pay their pay plan scheduled payments automatically, the account must be set up for automatic payment (as described under [How To Set Up A Taxpayer To Pay Automatically](#)). In addition, the pay plan must indicate that automatic payment is being used.

When this is done, a background process referred to as **C1-PAYPA** creates automatic payments on the scheduled payment date by calling the automatic payment creation algorithm plugged in on the installation record.



## Downloading Automatic Payments and Interfacing Them To The GL

The following diagram illustrates the background processes that interface automatic payment out of the system:



These processes are described in the following topics.

### Contents

[ACTVTAPY - Activating Automatic Payments](#)

[APAYACH - Download Automatic Payments To The ACH \(automated clearing house\)](#)

[BALAPY - Creating Automatic Payment Tender Controls](#)

### ACTVTAPY - Activating Automatic Payments

When an automatic payment is first created, it gets marked with an extract date. The extract date is the date the automatic payment will be downloaded to the respective financial institution. The extract date is determined as follows:

- Every automatic payment references an auto-pay source.
- Every auto-pay source references an auto-pay route type.
- Every auto-pay route type contains an algorithm that calculates this date.

On the extract date, the automatic payment is “activated”. The automatic payment is activated is marked for download the next time its download process runs. An automatic payment’s download process is defined on its auto-pay source’s route type.

### APAYACH - Download Automatic Payments To The ACH (automated clearing house)

This process reads all auto pay download staging records marked with a given batch control ID & run number and creates the flat file that’s passed to the ACH. Refer to [ACH Record Layouts](#) for the details of the record layouts.



**You can rerun this process.** You can reproduce the flat file at any time. Simply request this job and specify the run number associated with the historic run.

If you require a different flat file format, you must create additional versions of this program. Refer to [Setting Up Automatic Payment Extracts](#) for instructions describing how to add another process.

### BALAPY - Creating Automatic Payment Tender Controls

This process creates a new tender control (with an associated deposit control) for each unique batch control and run number encountered in the extracted automatic payments (where its payment tender is not yet linked to a tender control). The payment tender of each of these automatic payments is then linked to the corresponding tender control. This process also balances the open tender control records afterwards.

## ACH Record Layouts

The topics in this section describe the layout of the records created by [APAYACH – Download Automatic Payments To The ACH \(automated clearing house\)](#).

### Contents

- [File Header Record](#)
- [Company Batch Header Record](#)
- [Entry Detail Record](#)
- [Company Batch Control Record](#)
- [File Control Record](#)

### File Header Record

The ACH extract flat file must have one record of this type and it must be the first logical record on file.

Field Name	Format	Source/Value/Description
RECORD-TYPE	A1	"1"
PRIORITY-CD	A2	"01"
RESERVED-01	A1	Spaces
IMMEDIATE-DESTINATION	A9	CI_BANK_ACCOUNT.DFI_ID_NUM
COMPANY-ID	A10	CI_BANK_ACCOUNT.ACCOUNT_NBR
FILE-CRE-DT	A6	YYMMDD. Current date
FILE-CRE-TM	A4	HHMM. Current time
FILE-ID-MODIFIER	A1	"A"
RECORD-SIZE-CONST	N3	"094" The "FILLER" at the end of all but the Entry Detail Record serves to bring the total length of each record up to this constant.
BLOCKING-FACTOR	N2	"10"
FORMAT-CD	A1	"1"
ORIG-FIN-INST-NAME	A23	CI_BANK_L.DESCR
COMPANY-NAME	A23	CI_BANK_ACCOUNT_L.DESCR
REFERENCE-CD	A8	Spaces

**Company Batch Header Record**

The ACH extract flat file must have one record of this type and it must be the second logical record on file.

Field Name	Format	Source/Value/Description
RECORD-TYPE	A1	"5"
SERVICE-CLASS-CD	A3	"200"
COMPANY-NAME	A16	CI_BANK_ACCOUNT_L.DESCR
COMPANY-DISCRETIONARY	A20	Spaces
COMPANY-ID	A10	CI_BANK_ACCOUNT.ACCOUNT_NBR
STD-ENTRY-CLASS	A3	"PPD"
CO-ENTRY-DESCR	A10	"PAYMENT"
COMPANY-DESCR-DT	A6	Business process date
EFF-ENTRY-DT	A6	Business process date
RESERVED-01	A3	Spaces
ORIGINATOR-STAT-CD	A1	"1"
ORIGIN-DFI-ID	A8	CI_BANK_ACCOUNT.DFI_ID_NUM
BATCH-NBR	N7	The batch number of this batch within the file.

**Entry Detail Record**

The ACH extract flat file must have one record of this type for every direct debit record.

Field Name	Format	Source/Value/Description
RECORD-TYPE	A1	"6"
TRANSACTION-CD	A2	If the amount is a debit, this is set to CI_TENDER_TYPE.EXT_TYPE_FLG. If the amount is a credit, this is set based on the external type flag as follows: <ul style="list-style-type: none"> <li>If the value is Checking Withdrawal (27), this is set to 22 (Checking Deposit)</li> <li>If the value is Savings Withdrawal (37), this is set to 32 (Savings Deposit)</li> <li>If the value is Credit Card Withdrawal (47), this is set to 42 (Credit Card Deposit)</li> </ul>
TRANSIT-RTG-NBR	A9	CI_APAY_SRC.EXT_SOURCE_ID
DFI-ACCT-NBR	A17	CI_APAY_CLR_STG.EXT_ACCT_ID
AMOUNT	N8.2	CI_PAY_TNDR.TENDER_AMT
INDIVIDUAL-ID	A15	CI_PAY_TNDR.PAYOR_ACCT_ID
INDIVIDUAL-NAME	A22	CI_APAY_CLR_STG.ENTITY_NAME
DISCRETIONARY-DATA	A2	Spaces
ADDENDA-REC-IND	A1	"0"
TRACE-NBR	N15	"0000000000000000"

**External Account ID.** The EXT\_ACCT\_ID field supports up to 50 characters. If the value entered in the field is longer than that supported by the record layout (17 characters), the value will be truncated.

### Company Batch Control Record

The ACH extract flat file must have one record of this type and it must follow the Entry Detail records.

Field Name	Format	Source/Value/Description
RECORD-TYPE	A1	"8"
SERVICE-CLASS-CD	A3	"200"
ENTRY-ADDENDA-CNT	N6	The total number of entry detail records in this batch.
ENTRY-HASH	N10	The product of the first 8 digits of the external source id of the autopay source, multiplied by the number of entry detail records in the batch.
TOTAL-DR-DOLLAR-AMT	N10.2	Total tender amounts of the entry detail records.
TOTAL-CR-DOLLAR-AMT	N10.2	Zero.
COMPANY-ID	A10	CI_BANK_ACCOUNT.ACCOUNT_NBR
RESERVED-01	A19	Spaces
RESERVED-02	A6	Spaces
ORIGIN-DFI-ID	A8	CI_BANK_ACCOUNT.DFI_ID_NUM
BATCH-NBR	N7	The batch number of this batch within the file.

### File Control Record

The ACH extract flat file must have one record of this type and it must be the last logical record on file.

Field Name	Format	Source/Value/Description
RECORD-TYPE	A1	"9"
BATCH-CNT	N6	The number of batches in this file.
BLOCK-CNT	N6	Calculation of the total number of records in the file / 10.0 + 0.9
ENTRY-ADDENDA-CNT	N8	The total number of entry detail records in this file.
ENTRY-HASH	N10	The sum of the entry hash values on all the batch control records in this file.
TOTAL-DR-DOLLAR-AMT	N10.2	Sum of the total debit entry dollar amounts of all the batches in this file.
TOTAL-CR-DOLLAR-AMT	N10.2	Sum of the total credit entry dollar amounts of all the batches in this file.
RESERVED-01	A39	Spaces

## Issuing A Payment Advice Instead Of Creating An Automatic Payment

If the system is configured to send the taxpayer a payment advice (instead of initiating an electronic funds transfer) when a bill is completed, the automatic payment records – i.e. payment event, payment, tender and auto pay clearinghouse staging – are not created. Refer to [Payment Advices](#) for more information.

**Contents**

[How To Set Up A Taxpayer To Receive Payment Advices](#)  
[Payment Advice Option Is For Bill-Related Automatic Payments Only](#)

**How To Set Up A Taxpayer To Receive Payment Advices**

Use [Account – Auto Pay](#) to capture the taxpayer's bank details and indicate an auto pay method of **Payment Advice**.

**Payment Advice Option Is For Bill-Related Automatic Payments Only**

Payment advices can be printed for auto pays that result from completed bills.

An auto pay for a pay plan scheduled payment will not be created if the account's effective auto pay option is set to **Payment Advice**. The Pay Plan Pay Plan Auto Pay (**C1-PAYPA**) batch process will log an error in this case.

Manually created automatic payments (i.e. auto pays created via payment event UI) are always processed as direct debit.

## Maintaining Payment Events

A payment event is used to record when moneys are remitted and how the moneys are allocated amongst accounts. The topics in this section describe how to maintain payment events.

**The system creates most payment events behind-the-scenes.** Most payment events are created by the system when it [uploads payments](#) and when it creates [automatic payments](#). You should only have to access the payment event transaction if you need to correct a payment event or add a payment event real-time. For information about how the system creates payment events, refer to [The Big Picture of Payments](#).

**Contents**

[Payment Lifecycles](#)  
[Payment Event - Add Dialog](#)  
[Payment Event - Main Information](#)  
[Payment Event - Tenders](#)  
[Payment Event Action Codes](#)

## Payment Lifecycles

The topics in this section describe the lifecycle of the various payment objects.

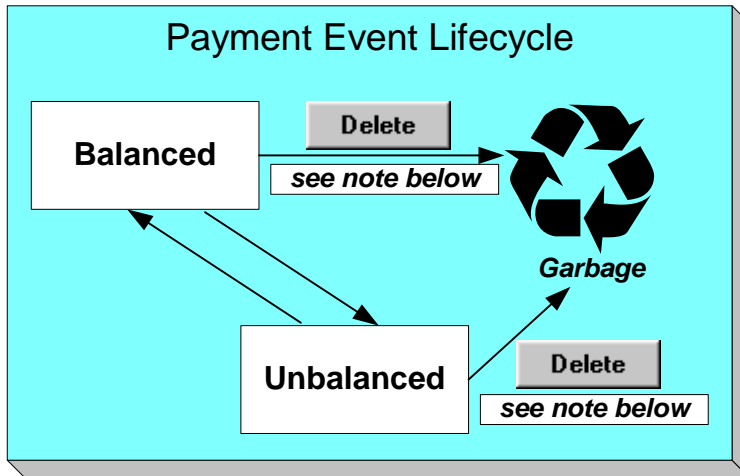
**Contents**

[Payment Event Lifecycle](#)  
[Tender Lifecycle](#)  
[Payment Lifecycle](#)

**Payment Event Lifecycle**

The following diagram shows the possible lifecycle of a payment event.

**Warning!** This diagram only makes sense in the context of the page used to maintain payment events. Refer to [Payment Event - Main Information](#) for the details.



The system, by default, distributes the sum of a payment event's tenders to the account that remits the tenders. After distribution the sum of the tenders equals the sum of the payments (remember, the term *payment* is used to refer to an allocation of some/all of a payment event's tenders to an account's debt) when the event is first created. We refer to such an event as being **Balanced**.

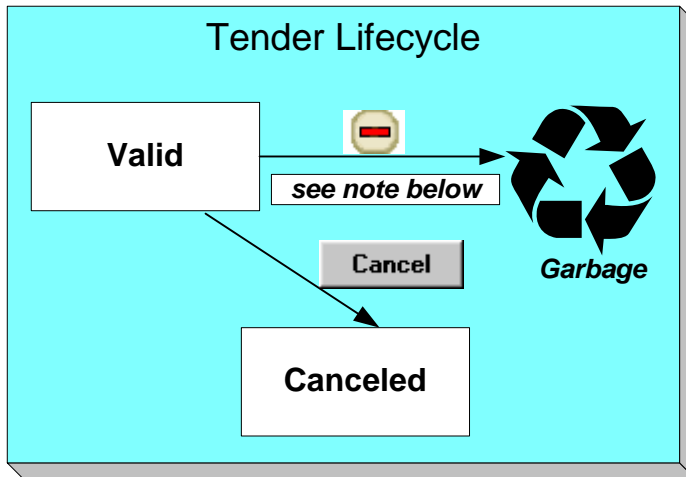
It is possible via any of the methods described in [Unbalanced Payment Events](#) to make a balanced payment event **Unbalanced**.

Click **Delete** to physically remove a balanced or unbalanced payment event from the database. You may not delete a payment event if: a) there are frozen or canceled payments linked to the event, or b) if there are canceled tenders linked to the event, or c) if a tender linked to the event is part of a balanced tender control. When the payment event is deleted, the system also deletes its tenders, payments, and payment segments.


## Tender Lifecycle

The following diagram shows the possible lifecycle of a payment event.

**Warning!** This diagram only makes sense in the context of the page used to maintain tenders. Refer to [Payment Event - Tenders](#) for the details.



A tender is initially saved in the **Valid** state. It is possible via any of the methods described in [Unbalanced Payment Events](#) to make a balanced payment event **Unbalanced**.

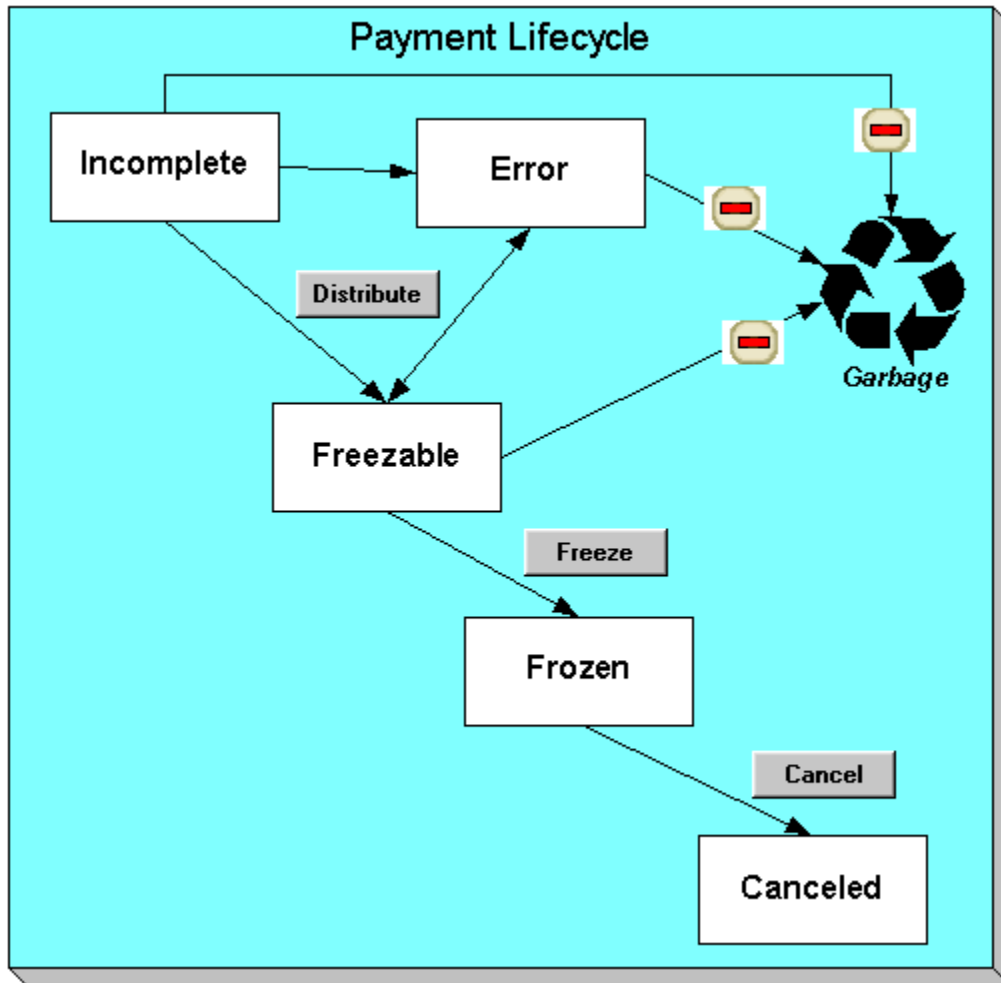
If a tender is invalid, click  to cancel the tender AND ALL PAYMENTS LINKED TO THE EVENT.

Click the delete button to physically remove a tender from the database. You may not delete a tender if it is part of a balanced tender control.

### Payment Lifecycle

The following diagram shows the possible lifecycle of a payment.

**Warning!** This diagram only makes sense in the context of the page used to maintain a payment. Refer to [Payment - Main](#) for the details.



Payments are initially created in the **Incomplete** state. Payments in this state don't have payment segments or financial transactions; they are simply a stub awaiting distribution.


Click **Distribute** to distribute a payment amongst an account's obligations.

- If the system cannot distribute the payment (for whatever reason), the payment is moved to the **Error** state. You may delete such a payment.
- If the system successfully distributes a payment, the payment becomes **Freezable**.

Click the - button to physically remove an **Incomplete**, **Error** or **Freezable** payment from the database.

Click **Freeze** to freeze a payment and its financial transaction. Freezing the payment causes the following to occur:

- The system executes any payment freeze algorithms linked to the account's [account type](#) and to the obligation's [obligation type](#).
- The payment's state becomes **Frozen** and the payment may now appear on a taxpayer's bill.

You may not change a payment once it is frozen. However, you may reverse the payment's financial effect by clicking  button. Clicking this button will cause the following to occur:

- A new financial transaction is generated and linked to the payment. This financial transaction reverses the financial effects of the original payment.
- The system executes any payment cancellation algorithms linked to the account's [account type](#).
- If the payment has a related adjustment, this adjustment is also cancelled.
- The payment becomes **Canceled**.

## Payment Event - Add Dialog

The Payment Event transaction features an unusual dialog that simplifies the addition of new payment events. This page appears if you open the **Financial, Payment Event** page in add mode from either the Account context menu or from the main menu (it also appears if you click the clear button when on the Payment Event page).

**Note.** If you have opted to always use the payment event distribution rules method as your [default method](#), the [Payment Event QuickAdd \(Single Payment Event\)](#) page appears instead.

### Description of Page

The **Payor Account ID** is the account that remitted the payment. We assume this account is both the tendering account and the account whose debt is being relieved by the payment. If this assumption is not correct, choose a **Distribute Action** of **Do Not Distribute** and then change the tendering or paying account when the Payment Event page appears (after you click **OK**).

**Default note.** If you have navigated to this page from account context menu, the Payor Account ID and Payment Amount are defaulted to this account.

The **Payment Amount** is the amount of the taxpayer's debt to be relieved by the payment. Note, this amount is defaulted using an algorithm plugged in on the [Installation Record](#). Please refer to [APAM-DFLT](#) for more information about how the algorithm that is supplied with the base package calculates this amount.

The payment tenders grid allows you to enter multiple tender types and amounts. Click + to add a new tender. For each tender specify the following fields:

- The **Amount Tendered** is the amount of moneys remitted for the tender type.
- **Tender Type** describes what was remitted (e.g., cash, check, ...). **Tender Type** defaults from the Quick Add Tender Type that is defined on the [installation record](#).
- If a check was tendered, use **Check Number** to specify the identity of the check.



**Cash back causes an additional tender to be created.** If cash should be returned to the taxpayer (because the taxpayer overpaid and the tender type's cash back allowed switch is true and the tender type is not "like cash"), a negative tender for the cash back amount is created for the payment event. Refer to [Cash Back](#) for a description of how the system can recommend Cash Back amount if the taxpayer tendered more than they are paying.

**Match Type** and **Match Value** are used if either of the following conditions is true:

- This **Payor Account** belongs to an [open item](#) account type. In this situation, specify a **Match Type** to define how the payment should be matched to the taxpayer's open-items and use **Match Value** to define the open-items covered by the payment. For example, if this payment is in respect of a bill, specify a match type of "bill id" and a match value of the bill id being paid.

**Shortcut.** If you enter a **Match Type** of "bill id" and leave the **Match Value** blank, the system assumes the taxpayer wants to pay the latest bill.

- The taxpayer wants to restrict the distribution of the payment to a specific obligation. In this situation, specify a **Match Type** of "obligation ID" and a **Match Value** of the respective obligation ID.

The **Payment Date** defaults to the current date.

If the **Payor Account ID's** [account type](#) is designated as non-tax agency payor (i.e., the person making the payment isn't a taxpayer), the following information appears in the above window:

**Non Tax Agency Name** is the name of the person remitting the payment.

**Reference Number** is the reference number of the item being paid (e.g., the property tax reference number).

**Non Tax Agency Comments** are used to describe anything unusual about the non tax agency payment.

Use **Distribute Action** to describe what you'd like to have happen when you click the **OK** button:

- Choose ***Distribute and Freeze if OK*** if this is a simple payment that should require no manual intervention. By "simple payment" we mean:
  - The account is both the tendering account and the account whose debt is being relieved by the payment
  - The payment date is the current date
  - The payment should be [distributed](#) amongst the account's obligations using standard distribution logic

If this option is selected, the system distributes the **Payment Amount** amongst the account's obligations. If the distribution is successful, the system automatically freezes the payment. If the distribution is not successful, the payment will be in the **Error** or **Incomplete** state. When the Payment Event page appears, you can view the error and then correct it. After the cause of the error is corrected, you must distribute and freeze the payment manually (this can be done on several pages including [Payment Event - Main](#) and [Payment - Main](#)).

- Choose **Manual Distribution** if you need to manually distribute the payment amongst the account's obligations. If this option is selected, the system creates a payment event, a tender and a payment and then transfers you to the [Payment - Manual Distribution](#) page where you can define the amount to be allocated to each of the account's obligations. After you've distributed the payment, don't forget to freeze it.
- Choose **Do Not Distribute** if you want to process the payment event manually (e.g., if you need to define multiple accounts whose debt is relieved by the payment). If this option is selected, the [Payment Event – Main](#) page opens with the information you entered defaulted accordingly. You can make any changes you want and then distribute and freeze the payment. Refer to [How To Add A New Payment Event](#) for more information.

## Payment Event - Main Information

The Main page contains core payment event information. Open this page using **Financial, Payment Event**.

The Description of Page section that appears below simply describes the fields on this page. Refer to [How To](#) for a description of how to perform common payment event maintenance functions.

### Description of Page

**Pay Event Info** contains a concatenation of the payment date, amount, and the name of the main taxpayer on the account that remits the tender. If multiple tenders exist, the taxpayer's name is not displayed. If the payment event is associated with a single distribution detail, the rule name and the description of the rule value are displayed as well. If multiple distribution details exist, **Multiple Distribution Details Exist** is displayed instead. **Pay Event Info** is only displayed after the payment event has been added to the database.

**Payment Event ID** is the system-assigned unique identifier of the payment event.

The area under **Pay Event Info** provides warnings about the payment event. Possible warnings are **Unfrozen Payments** and **The Payment Event is Unbalanced**. A warning is also issued to the cashier to remind him/her to turn in funds – see [Turn Ins](#).

If multiple distribution details are linked to the payment event, the distribution rule, value and amount for each distribution detail is displayed.

**Payment Date** is the business date associated with the payment event.

**Warning!** If you change the payment date and this event's tender(s) are automatic payments, the extract date (i.e., the date the automatic payment is sent to the financial institution) will be changed to equal the payment date. Refer to [Automatic Payments](#) for more information.

**Effective Date** is populated by the algorithm that creates the payment and is used to populate the FT effective date. For example, an income tax return that the taxpayer files on April 1<sup>st</sup> that is due on April 15<sup>th</sup> has an effective date of April 15<sup>th</sup>.

**Changing effective date.** Refer to [Payment Date and Effective Date for Payment Events](#) for information about changing the effective date of a distributed payment event.

**Payment(s)** contains the total number and value of payments linked to the event.

**Tender(s)** contains the total number and value of tenders linked to the event.

**Amount Tendered** contains the amount that was tendered by the taxpayer.

The system displays a **Cash Back Amount** if the taxpayer tenders more than they are paying (refer to [Cash Back](#) for details).

If the payment event becomes **unbalanced**, a **Recalculate Cash Back** button appears. If this button is clicked, the system creates, deletes or recalculates the cash back tender. Refer to [Cash Back](#) for details.

While most payment events contain a single payment, the system allows many payments to exist under a payment event (when the payment event's tender(s) are distributed amongst multiple accounts). If a payment event has a large number of payments, you can use the **Account Filter** to limit the payments that appear in the **Payments** grid. The following options are available:

- **Account.** Use this option if you only want to see the payment linked to an **Account ID**.
- **All.** Use this option to view all payments linked to the payment event.
- **Person Name.** Use this option to restrict payments linked to accounts whose main taxpayer has a primary name that matches **Person Name**.

The **Payments** grid contains the payment(s) linked to the payment event. The following points describe the attributes in this scroll; refer to [How To](#) for instructions describing how to perform common maintenance activities.

- If the payment is created via distribution details, the **Distribution Sequence** associated with this payment is displayed.
- **Account ID** references the payment's account. The name of the account's main taxpayer is displayed adjacent.
- **Payment Amount** is the amount of the account's debt relieved by the payment. The adjacent context menu allows you to drill down to the details of the [payment](#) (this is where you can see the payment's payment segments and where you can override the distribution of the payment).
- **Payment Status** is the payment's status. If the payment's status is **Error**, the error message is displayed adjacent. Refer to [Payment Lifecycle](#) for the potential values and how to handle a payment when it exists in a given state.
- **Match Type** and **Match Value** should only be used if either of the following conditions is true:
  - This **Account ID** belongs to an [open item](#) account type. In this situation, specify a **Match Type** to define how the payment should be matched to the taxpayer's open-items and use **Match Value** to define the open-items covered by the payment. For example, if this payment is in respect of a bill, specify a match type of "bill id" and a match value of the bill id being paid.
  - The taxpayer wants to restrict the distribution of the payment to a specific obligation. In this situation, specify a **Match Type** of "obligation ID" and a **Match Value** of the respective obligation ID.
- The remaining columns are only used if the payment is linked to an **Account ID** that belongs to an account type that is used for non-CIS payments. Refer to [Setting Up Account Types](#) for more information. If such an account exists, the following fields must be defined.
  - **Non Tax Agency Name** is the name of the person remitting the payment.

- **Reference Number** is the reference number of the item being paid (e.g., the property tax reference number).
- **Non Tax Agency Comments** are used to describe anything unusual about the non tax agency payment.

Note, you can also define a comment for a non tax agency payment. To define a comment, use the context menu adjacent to **Payment Amount** to drill to [Payment - Main](#).

- **Payment ID** is the unique, system-assigned identifier of the payment. This value only appears after the payment has been added to the database.

Refer to [Payment Actions](#) and [Payment Event Actions](#) for information about the action buttons on this page. Refer to [How To](#) for a description of typical business processes that use these buttons.

## Payment Event - Tenders

The Tenders page contains a scroll showing one row for every tender associated with the payment event. Open this page using **Financial, Payment Event, Tenders**.

The Description of Page section that appears below simply describes the fields on this page. Refer to [How To](#) for instructions describing how to perform common maintenance functions.

### Description of Page

**Pay Event Info** contains a concatenation of the payment date, amount, and the name of the main taxpayer on the account that remits the tender. If multiple tenders exist, the taxpayer's name is not displayed. If the payment event is associated with a single distribution detail, the rule name and the description of the rule value are displayed as well. If multiple distribution details exist, **Multiple Distribution Details Exist** is displayed instead. **Pay Event Info** is only displayed after the payment event has been added to the database.

**Payment Event ID** is the system-assigned unique identifier of the payment event.

**Payment Date** is the business date associated with the payment event.

**Payment(s)** contains the total number and value of payments linked to the event.

**Tender(s)** contains the total number and value of tenders linked to the event.

**Amount Tendered** contains the amount that was tendered by the taxpayer.

The system displays a **Cash Back Amount** if the taxpayer tenders more than they are paying (refer to [Cash Back](#) for details).

If the payment event becomes **unbalanced**, a **Recalculate Cash Back** button appears. If this button is clicked, the system creates, deletes or recalculates the cash back tender. Refer to [Cash Back](#) for details.

The **Tenders** scroll controls the display of the tenders linked to the payment event. The following simply describes the fields in this scroll; refer to [How To](#) for instructions describing how to perform common tender maintenance activities.

- **Payor Account ID** references the tendering account. The name of the account's main taxpayer is displayed adjacent.

**Warning!** If you change the Payor Account ID and the tender has an associated automatic payment request, the automatic payment request will be removed. A new automatic payment request will be created for the new Payor Account ID. Refer to [Automatic Payments](#) for more information.

- **Tender Amount** is the amount of the tender.
- If a check was used, the **Check Number** contains the identity of the check.
- **Pay Tender ID** is the system-assigned unique identifier of the tender. This value appears after the tender has been added to the database.
- **Tender Type** describes what was remitted (e.g., cash, check). If the **Tender Type** is associated with an automatic payment, the page displays a section that includes information about how and when the automatic payment was interfaced to the payment source.
- **Tender Status** is the tender's status. Refer to [Tender Lifecycle](#) for the potential values and how to handle a tender when it exists in a given state.
- If the **Tender Type** is associated with an [automatic payment](#), the system attempts to default automatic payment information from the [account's auto-pay](#) option if the tender type is the same as the tender type on the account's auto-pay source and if the auto pay option is effective on the payment date. If the system is unable to default information, you must specify the source of the funds and the taxpayer's account number / credit card number at the financial institution. You can override automatic payment information as long as the automatic payment has not been sent to the financial institution or cancelled.
  - **Auto Pay Source Code** is the financial institution / credit card company that receives the automatic payment request.
  - **Scheduled Extract Date** is the date the automatic payment request was sent / is scheduled to be sent to the financial institution. This information is display-only.
  - **External Account ID** is the taxpayer's account number at the financial institution.
  - **Expires On** is only needed if the Tender Type indicates that an expiration date is necessary (e.g., for a credit card payment).
  - **Name** is the taxpayer's name in the financial institution's system.
  - **Bill ID** is the ID of the bill whose completion caused the creation of the automatic payment. This information only appears if the automatic payment was created as a result of a bill. This information is display-only.
- If the payment was uploaded, the following fields MAY contain information (if they were populated in the interface file):
  - **MICR ID** is the value of the magnetic ink character recognition (MICR) line on the payment.
  - **Taxpayer ID** is the taxpayer's account ID that appeared on the interfaced payment.
  - **Ext. Reference ID** is the unique identifier of the payment upload interface record.
  - **Name** is the taxpayer's name on the interfaced payment record.
- In the **Tender Control** area, the **Tender Control ID** is the identity of the remittance processor batch or cash drawer bundle in which the tender was remitted. Refer to [Tender Management and Workstation Cashiering](#) for more information about tender controls.

Displayed adjacent to **Tender Control** is the date, tender source, and status of the tender control.

Every new tender must reference an **open** tender control. The system will attempt to default an appropriate tender control as follows:

- If the user has at least one **open**, user-specific Tender Control, the system will default one, at random.
- If the user has no **open**, user-specific Tender Controls; the system will default an open, “all users” tender control if it can find one whose tender source indicates it’s used for online cashiering. If multiple exist, one will be selected at random.

Turn off **Included in Tender Ctl Balance** if the tender should not be included in the tender amount in the tender control’s tender balance. You would turn this switch off if you canceled a tender because it was mistakenly applied to the wrong account (or the amount was wrong). This switch is only enabled if:

- The tender control isn’t **balanced**.
- The tender is **canceled**.

**Deposit Control** contains a concatenation of the tender control’s deposit control’s creation date, tender source type, and status. Refer to [Tender Management and Workstation Cashiering](#) for more information about deposit controls.

- The **Tender Action** area contains a button that you use to cancel a tender. Refer to [Tender Actions](#) for information about this button. Refer to [How To Cancel A Tender](#) for more information.

The **Characteristics** collection contains information that describes miscellaneous information about the tender. The following fields display:

<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Sequence</b>	Controls the order in which characteristics of the same type are displayed.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

## Payment Event Action Codes

The topics that follow describe each of the actions that appear on the payment event pages.

### Contents

[Payment Actions](#)  
[Payment Event Actions](#)  
[Tender Actions](#)

### Payment Actions

The topics that follow describe payment-oriented actions. Refer to [How To](#) for a description of several business processes that use these buttons.

### Contents

[Distribute \(Payments\)](#)  
[Redistribute \(Payments\)](#)  
[Print](#)

### Freeze (Payments)

#### Distribute (Payments)

Clicking the **Distribute** button distributes all **Incomplete**, **Error**, or **Freezable** payments linked to the event. Refer to [Distribute \(A Payment\)](#) for information of how a single payment is distributed.

This button is enabled if at least one payment is **Incomplete**, **Error**, or **Freezable**.

#### Redistribute (Payments)

Clicking the **Redistribute** button redistributes all **Incomplete**, **Error**, or **Freezable** payments linked to the event. Refer to [Redistribute \(A Payment\)](#) for information of how a single payment is redistributed.

This button is enabled if at least one payment is **Incomplete**, **Error**, or **Freezable**.

#### Print

This button is enabled under the following conditions:

- The document **cm\_POSPrint.js** must be present on the web server and the variable defined in that document must be set to **true**. This is to indicate that point of sale (POS) printers are in use. Refer to the installation guide for more information about setting up receipt printers.
- The payment event must have at least one payment that is **frozen** AND no payments that are in **Incomplete**, **Error** or **Freezable**

Refer to [How To Print Receipts And Endorsements](#) for more information.

#### Freeze (Payments)

Clicking the **Freeze** button causes all **Freezable** payments linked to the event to become **frozen**. Refer to [Freeze \(A Payment\)](#) for information of how a single payment is frozen.

This button is enabled if at least one payment is **Freezable**.

**If problems are detected after freezing.** A payment may not be changed after it is **frozen**. All subsequent changes must occur by canceling the frozen payment and creating a new one. You can cancel and redistribute a payment on the next tab. Refer to [Canceling A Tender Versus Canceling a Payment](#) for more information.

## Payment Event Actions

The topics that follow describe payment event-oriented actions. Refer to [How To](#) for a description of typical business processes that use these buttons.

### Contents

[Transfer](#)

[Delete A Payment Event](#)

#### Transfer

If the payment event was created using distribution rule(s), any transfer of its payments should also be done using the distribution rules method. Hence, a different dialog opens when you click **Transfer** based on whether the payment event is associated with at least one distribution detail or not.



## Contents

[Transfer \(No Distribution Details Exist\)](#)

[Transfer \(Distribution Details Exist\)](#)

### [Transfer \(No Distribution Details Exist\)](#)

The **Transfer** button is enabled if:

- There is a single non-cancelled tender linked to the event AND
- There is a single frozen payment linked to the event AND
- The account on the tender is the same as the account on the payment.

You must specify the following parameters in order to transfer a payment:

- **Account ID** is the account to which the payment event should be transferred.
- **Cancel Reason** defines why you are performing the transfer.
- **Match Type** and **Match Value** are defaulted based on their values for the payment event that you are transferring. If necessary, modify the values appropriately for the account to which you are transferring the payment event. Match Type and Match Value are used for an account that belongs to an [open item](#) account type or to restrict the distribution of the payment to a specific obligation. Refer to [Payment Event - Main](#) for more information.
- Turn on **Freeze Payment** if the transferred payment should be frozen automatically. You may want to leave this turned off if you want to examine the payment distribution before freezing the payment.
- **Non Tax Agency Name**, **Reference Number**, and **Non Tax Agency Comments** appear if the **Account ID** belongs to an account type that is used for non tax agency payments. Refer to [Setting Up Account Types](#) for more information.

Clicking **OK** causes the following to take place:

- The account on the tender is changed to reflect the transfer to account.
- The original payment and its segments are cancelled.
- A new payment is added for the transfer to account. The payment characteristics associated with the original payment are copied to the new payment.
- The new payment is distributed.
- If so designated, the new payment is frozen.

### [Transfer \(Distribution Details Exist\)](#)

The **Transfer** button is enabled if:

- There is a single payor AND
- All existing payees before the transfer are the same as the payor AND
- There is at least a single non-cancelled tender linked to the event

You must specify the following parameters in order to transfer payment(s):

- **Distribution Rule** is the account to which the payment event should be transferred.
- **Rule Value** is the value associated with the payment and expected by the distribution rule.



- **Amount** is the payment amount.
- **Cancel Reason** defines why you are performing the transfer.

Clicking **OK** causes the following to take place:

- The payment(s) associated with the original account are canceled.
- The new set of distribution details are processed, creating new payment(s) for the transfer to account. The payment characteristics associated with the original payment are copied to the new payment.
- If all new payments are for the same payee account (and the payee is different than the current payor) the account on the tender is changed to reflect the transfer to account.
- Original distribution details (only those with a non-zero amount) are deleted and replaced by the new set of details on the payment event.

**Distribution rule entries can have a zero amount.** Refer to [Rule Value Can Capture Additional Information](#) for more information about how to use distribution details to capture additional payment related information.

**Determine Tender Account.** This dialog does not call the *Determine Tender Account* algorithm defined on [distribution rule](#) as this action does not rebuild the payment tender collection.

Refer to [Rule Value Can Capture Additional Information](#) for more information about how to use distribution details to capture additional payment related information.

### Delete A Payment Event

The **Delete** button deletes a payment event and its tenders and payments.

This button is enabled if there are no *frozen* or *canceled* payments and no *canceled* tenders.

## Tender Actions

The topics that follow describe tender-oriented actions. Refer to [How To](#) for a description of several business processes that use these buttons.

### Cancel A Tender

Clicking **Cancel** causes the following to take place:

- All payments associated with the tender's payment event are *canceled*.
- The tender becomes *canceled*.
- If the cancellation reason indicates the cancellation is due to non-sufficient funds, other actions may occur. Refer to [NSF Cancellations](#) for more information.

The **Cancel** button is enabled if the tender is not *canceled*

You must specify the following parameters in order to cancel a tender:

- Select a **Cancel Reason** to describe why the tender is being canceled.
- The system also needs to know which **Bank Account To Charge** the cancellation against. The system will default the **Bank Code** and **Bank Account** from the original bank information used when the tender was deposited. You can override them here.

When you click the **OK** button, the system cancels the tender and ALL payments linked to the event.

**When non-canceled tenders are still linked to the event.** If there are multiple tenders linked to the payment event, you must either add new payment(s) that equal the sum of the event's non-canceled tenders or cancel the remaining tenders.

**Cancellation after cash back.** If a taxpayer tenders a check for \$100, but only owes \$80, the system will recommend returning \$20 in cash to the taxpayer (assuming the tender type for checks allows overpayment). If the tender type for checks is not "like cash", a second tender is created for -\$20 (the first tender is for \$100). If the check subsequently bounces, both tenders must be cancelled.

## Payment Event QuickAdd

The Payment Event QuickAdd page is used to quickly add, distribute and freeze one or more payment events using [distribution rules](#). Open this page using **Financial, Payment Event QuickAdd**.

### Description of Page

Specify the **Tender Control ID** in which the tenders will be recorded. Every new tender must reference an **open** tender control. The system will attempt to default an appropriate tender control as follows:

- If the user has at least one **open**, user-specific Tender Control whose type is online cashiering or ad hoc, the system will default one, at random.
- If the user has no **open**, user-specific Tender Controls; the system will default an open, "all users" tender control if it can find one whose tender source whose type is online cashiering or ad hoc. If multiple exist, one will be selected at random.

Specify the **Payment Date**. The current date defaults. Note that you can modify the payment effective date on the Payment Event page. See [Payment Event - Main](#) for more information.

Use **Number of Payment Events** to indicate whether you intend to add a **single** payment event or **multiple** payment events.

Choose the **multiple** payment events option if you intend to add "simple" payment events. By "simple" we mean:

- A single account is both the payor account and the account whose debt is being relieved by the payment.
- The payment tender is not an auto-pay tender.

For all other cases choose the **single** payment event dialogue as it is designed to handle more complex payment event configurations such as when:

- The payor account is different than the payee account.
- Multiple payors cover payments for one or more payees.
- The payment tender is an auto-pay tender.

Once you have made your selection the page is adjusted to support the corresponding dialog.

## Contents

[Multiple Payment Events Dialog](#)

[Single Payment Event Dialog](#)

## Multiple Payment Events Dialog

This is the default dialog setup when navigating from **Financial, Payment Event QuickAdd**.

### Description of Page

Using the **multiple** payment events option, each row in the payment detail grid typically represents a unique distribution detail. However, more than one row in the grid can belong to a single payment event. All distribution details for identical tender account will be grouped under a single pay event.

The following columns appear in the **Payments** grid:

- Select a **Distribution Rule** by which the payment detail is to be processed. If you have set up a [default distribution rule](#), it is defaulted in the first row.
- Specify the **Rule Value** associated with the payment and expected by the distribution rule.
- Use **Tender Amount** to define the amount of the payment.
- Use **Tender Type** to define the form or remittance (e.g., cash, check, etc.). Note that the **Tender Type** defaults from the [installation record](#).

**Automatic Payments.** The multiple payment events dialog does not support automatic payments. Switch to the single payment event dialog to enter automatic payments.

- Use **Check Number** if a check is remitted.
- **MICR ID** is the value of the magnetic ink character recognition (MICR) line on the payment.
- You may use **Customer ID** to record additional taxpayer information.
- You may use **External Reference ID** to record external information associated with the payment tender.
- You may use **Name** to record additional payment tender information.
- **Payor Account ID** is the tender account as determined by the [Determine Tender Account](#) algorithm defined on the [distribution rule](#). This information is only populated after the distribution detail has been processed.

After specifying the various payment distribution details in the grid, click **Create**. The system attempts to create payment event(s) as follows:

- As mentioned earlier, all distribution details for identical tender accounts are grouped under a single pay event. The first step in identifying common tenders is to determine the tender account. To do that the system calls the **Determine Tender Account** algorithm defined on the [distribution rule](#).
- A payment event is created for each unique tender account.
- All rows having all attributes of the tender identical except for the amount are grouped together where each group represents a single tender.
- A single payment tender is created for each unique tender and linked to the payment event.
- The total payment amount for each distinct group of distribution details (for the payment event) having the same rule and value is summarized. For each distinct group, the system calls the **Create Payment** algorithm defined on the [distribution rule](#) providing it with the rule value and total payment amount. It is the responsibility of this algorithm to create the payments for this payment event.
- Add the distinct payment event distribution detail and its total amount beneath the payment event.
- If the tender control's tender source type indicates that this is a cashiering station and your implementation is configured for printing at a cashier station, the [Print Dialog](#) appears.

The **Payment Event ID** column appears in the grid after the distribution details have been processed. It shows the payment event ID created for the distribution detail record and a short description as to the status of the payment(s) created for the payment event. You can use this field to navigate to the payment event.

After you've added a group of payment events, you should press the [clear](#) button (or Alt-C), to ready the page for the next group of payment event details.

## Single Payment Event Dialog

If you have opted to always use the payment event distribution rules method as your [default method](#), this is the default dialog setup you would see when you navigate to the Payment Event page in Add mode.

Adding a **single** payment event at a time, allows for more complex payment event information to be captured.

Enter one row in the **Tenders** grid for every unique tender associated with the payment event.

- **Payor Account ID** references the tendering account. The name of the account's main taxpayer is displayed adjacent.

**Warning!** If you change the Payor Account ID and the tender has an associated automatic payment request, the automatic payment request will be removed. A new automatic payment request will be created for the new Payor Account ID. Refer to [Automatic Payments](#) for more information.

- **Tender Amount** is the amount of the tender.
- Use **Tender Type** to define the form or remittance (e.g., cash, check, etc.). Note, the **Tender Type** defaults from the [installation record](#).
- Use **Check Number** if a check is remitted.

- **MICR ID** is the value of the magnetic ink character recognition (MICR) line on the payment.
- You may use **Customer ID** to record additional taxpayer information.
- You may use **External Reference ID** to record external information associated with the payment tender.
- You may use **Name** to record additional payment tender information.
- If the **Tender Type** is associated with an automatic payment, the page displays a section that includes information about how and when the automatic payment was interfaced to the payment source.
- If the **Tender Type** is associated with an [automatic payment](#), the system attempts to default automatic payment information from the [account's auto-pay](#) option if the tender type is the same as the tender type on the account's auto-pay source and if the auto pay option is effective on the payment date. If the system is unable to default information, you must specify the source of the funds and the taxpayer's account number / credit card number at the financial institution.
  - **Auto Pay Source Code** is the financial institution / credit card company that receives the automatic payment request.
  - **External Account ID** is the taxpayer's account number at the financial institution.
  - **Expires On** is only needed if the **Tender Type** indicates that an expiration date is necessary (e.g., for a credit card payment).
  - **Name** is the taxpayer's name in the financial institution's system.

The following columns appear in the **Payments** grid:

- Select a **Distribution Rule** by which the payment detail is to be processed. If you have set up a [default distribution rule](#), it is defaulted in the first row.
- Specify the **Rule Value** associated with the payment and expected by the distribution rule.
- Use **Amount** to define the amount of the payment.

After specifying the various payment distribution details in the grid, click **Create**.

**Go To Payment Event.** If you wish to be transferred to the payment event page once processing is complete, remember to check the Go To Payment Event check box before you click **Create**. This field does not appear when this page is used as the default Payment Event - Add Dialog and you are automatically transferred to the payment event page once the payment event is created.

After distribution details have been processed, a short description of the payment event that has been created is displayed to the right of the **Go To Payment Event** field. The description provides information as to the status of the payment(s) created for the payment event. You can use this field to navigate to the payment event.

The system attempts to create the payment event as follows:

- Validate that the sum of the tenders equals the sum of the payments.
- Create the payment event with a separate payment tender for each row in the Tenders grid.

- Summarize the total payment amount for each distinct group of distribution details having the same rule and value. For each distinct group, call the **Create Payment** algorithm defined on the [distribution rule](#) providing it with the rule value and total payment amount. It is the responsibility of this algorithm to create the payments for the payment event.
- Add the distinct payment event distribution detail summary beneath the payment event.
- If the tender control's tender source type indicates that this is a cashiering station and your implementation is configured for printing at a cashier station, the [Print Dialog](#) appears.

**Determine Tender Account.** Having an explicit collection of tenders eliminate the need to determine the tender account. Therefore this dialog does not call the **Determine Tender Account** algorithm.

## Payment QuickAdd

The Payment QuickAdd page is used to quickly add, distribute and freeze payment events for up to fifteen accounts. Open this page using **Financial, Payment QuickAdd**.

### Description of Page

Specify the **Tender Control ID** in which the tenders will be recorded. Every new tender must reference an **open** tender control. The system will attempt to default an appropriate tender control as follows:

- If the user has at least one **open**, user-specific Tender Control whose type is online cashiering or ad hoc, the system will default one, at random.
- If the user has no **open**, user-specific Tender Controls; the system will default an open, "all users" tender control if it can find one whose tender source whose type is online cashiering or ad hoc. If multiple exist, one will be selected at random.

Specify the **Payment Date**. The current date defaults. Note that you can modify the payment effective date on the Payment Event page. See [Payment Event - Main](#) for more information.

Enter one row in the grid for every payment event to be added. The following information is entered for each payment event:

- Use **Account ID** to define the taxpayer who tendered the payment. It's important to note that this transaction assumes that the tendering account is the same as the account whose balance is relieved by the payment. Refer to [How To Allocate The Tender Amount To Multiple Accounts](#) if you need to distribute the payment to an account (or accounts) other than the tendering account.
- Use **Payment Amount** to define the amount of the payment.
- Use **Tender Type** to define the form or remittance (e.g., cash, check, etc.). Note, the **Tender Type** defaults from the [installation record](#).
- Use **Check Number** if a check is remitted.
- **Match Type** and **Match Value** are only used if either of the following conditions is true:

- This **Account ID** belongs to an [open item](#) account type. In this situation, specify a **Match Type** to define how the payment should be matched to the taxpayer's open-items and use **Match Value** to define the open-items covered by the payment. For example, if this payment is in respect of a bill, specify a match type of "bill id" and a match value of the bill id being paid.

**Shortcut.** If you enter a **Match Type** of "bill id" and leave the **Match Value** blank, we assume the taxpayer wants to pay the latest bill.

- The taxpayer wants to restrict the distribution of the payment to a specific obligation. In this situation, specify a **Match Type** of "obligation ID" and a **Match Value** of the respective obligation ID.

After specifying the various accounts and amounts in the grid, click the **Distribute and Freeze** button. When you click this button, the system attempts to create a payment event for each row in the grid. Four potential outcomes are possible for each row:

- If the data you entered for a payment event isn't complete (e.g., you don't specify a valid account or amount):
  - No payment event is created.
  - A **Message** describing the problem is displayed.
  - All fields on the row remain modifiable.

You should correct each such line and then press the **Distribute and Freeze** button.

- If the data you entered is complete, but the system issues a warning for any reason:
  - No payment event is created.
  - A **Message** containing the warning is displayed.
  - All fields on the row remain modifiable.

You must clear the line and add a payment using the [Payment Event transaction](#) (note, if you use the Account context menu to transfer to the Payment Event page in add mode, you won't have to retype the Account ID when you add the payment event).

- If the data you entered is complete, but the system is not successful in distributing the payment:
  - The system creates a payment event, a payment and a tender.
  - A **Message** describing the problem is displayed.
  - The payment's **Status** is displayed
  - All fields on the row are disabled.

You must press the adjacent go to button to drill to the payment event where the error can be corrected.

- If the data you entered is complete and the system is successful in distributing the payment:
  - The system creates a payment event, a payment, and a tender.
  - The system distributes the payment amongst the account's obligations and then freezes the payment.
  - The payment's **Status** is displayed.

- All fields on the row are disabled.
- If the tender control's tender source type indicates that this is a cashiering station and your implementation is configured for printing at a cashier station, the [Print Dialog](#) appears.
- You can press the adjacent go to button to view a payment event.

**Separate Commits.** This page is unusual in that each payment event is committed to the database independently. For example, if you enter seven payment events and one is invalid, six payment events will be added to the database when you press **Distribute and Freeze**. When the page is redisplayed, the rows containing the valid payments are protected and an indication of their validity is displayed. The row containing the invalid payment remains unprotected. You can correct the erroneous payment and then press the **Distribute and Freeze** button again.

After you've added a group of payments, you should press the [clear button](#) (or Alt-C), to ready the page for the next group of payment events.

## Maintaining Payments

Payments and payment segments are automatically created by the system when [payment events](#) are created. You should only need to access this transaction as follows:

- To view a payment's payment segments (i.e., to view how a payment was distributed amongst its account's obligations).
- To override the distribution of a payment amongst its account's obligations. Refer to [Distributing A Payment Amongst An Account's Obligations](#) for information describing how payments are typically distributed.

The topics in this section describe how to maintain a payment.

### Contents

[Payment - Main](#)  
[Payment - Pay Segments](#)  
[Payment - Manual Distribution](#)  
[Payment - Characteristics](#)  
[Payment Action Codes](#)

## Payment - Main

This page contains basic information about a payment. Open this page using **Financial, Payment, Main**.

Most payments are maintained using the [Payment Event](#) page. The transaction described below is typically only used to view a payment's payment segments and to manually distribute a payment amongst an account's obligations.

The Description of Page section that appears below simply describes the fields on this page. Refer to [How To](#) for instructions on how to perform common maintenance functions.



### Description of Page

**Payment Info** contains a concatenation of the payment date, amount, status, and the name of the main taxpayer on the account.

**Payment ID** is the system-assigned unique identifier of the payment.

**Payment Event ID** is the system-assigned unique identifier of the payment event. The adjacent info contains a concatenation of the payment date, amount, and the name of the main taxpayer on the account that remits the event's tender. If multiple tenders exist, the taxpayer's name is not displayed. A warning may also be displayed. The possible warnings are *Unfrozen Payments* and *The Payment Event is unbalanced*.

**Payment(s)** contains the total number and value of payments linked to the payment event.

**Tender(s)** contains the total number and value of tenders linked to the payment event.

**Account ID** is the account whose debt is reduced by the payment. The name of the account's main taxpayer is displayed adjacent. This field is gray after the payment is frozen.

**Payment Amount** is the amount of the payment. This field is gray after the payment is frozen.

**Payment Status** is the payment's status. Refer to [Payment Lifecycle](#) for the potential values and how to handle a payment when it exists in a given state. If the payment is in *Error*, the error message is displayed adjacent. If the payment is *canceled*, the **Cancel Reason** is displayed adjacent.

**Match Type** and **Match Value** will only be used if either of the following conditions is true:

- This **Account ID** belongs to an [open item](#) account type. In this situation, specify a **Match Type** to define how the payment should be matched to the taxpayer's open-items and use **Match Value** to define the open-items covered by the payment. For example, if this payment is in respect of a bill, specify a match type of "bill id" and a match value of the bill id being paid.
- The taxpayer wants to restrict the distribution of the payment to a specific obligation. In this situation, specify a **Match Type** of "obligation ID" and a **Match Value** of the respective obligation ID.

**Match Type** and **Match Value** are gray after the payment is frozen.

If the payment is linked to an **Account ID** that belongs to a [account type](#) that is used for non tax agency payments, the following fields appear (note, these fields are gray after the payment is frozen):

- **Name** is the name of the person remitting the payment.
- **Reference Number** is the reference number of the item being paid (e.g., the property tax reference number).
- **Comments** are used to describe anything unusual about the non tax agency payment.

The contents of the **Payment Segments** section depends on the number of payment segments linked to the payment. If more than 25 payment segments exist, a message appears that allows you to navigate to [Payment - Payment Segments](#) where you can view the payment segments (this tab page has special functionality that allows you to define which payment segments should be displayed). If 25 or fewer payment segments exist, the grid contains information about each payment segment:

- **Location** is the address of the obligation's main premise.

- Click the go to button to transfer to view the financial transaction related to the payment segment. This is only available after the payment has been frozen.
- **Obligation Information** contains basic information about the obligation.
- **Distributed Amt** is the amount of the obligation's debt relieved by the payment.

The following information appears at the bottom of the page:

- **Payment Amount** is the amount of the payment.
- **Distributed Amount** is the sum of the payment segments linked to the payment.
- The **Difference** between the **Payment Amount** and the **Distributed Amount** appears if it is non-zero.
- **Billed Amount** is the amount of the bill preceding the payment date.
- **Delinquent Amount** is the amount of the taxpayer's debt that was due on / before the prior bill's due date.
- **Current Balance** is the account's current balance.

Refer to [Payment Action Codes](#) for information about the action buttons on this page.

## Payment - Pay Segments

You can use this page to view all or selected payment segments linked to a payment.

Open **Financial, Payment, Payment Segments** to view this information.

Most payments are maintained using the [Payment Event](#) page. The transaction described below is typically only used to view a payment's payment segments.

**Note.** If the payment has more than 25 payment segments, the search criteria are intentionally left blank in order to avoid retrieving all payment segments (with the resultant slow response times). You must therefore use the **Obligation Filter** to define the type of payment segments that should be retrieved. See the [Description of page](#) below for more information about this page's search criteria.

### Description of page

**Payment Info** contains a concatenation of important information about the payment.

**Payment ID** is the system-assigned unique identifier of the payment.

**Payment Event ID** is the system-assigned unique identifier of the payment event. The adjacent info contains a concatenation of the payment date, amount, and the name of the main taxpayer on the account that remits the event's tender. If multiple tenders exist, the taxpayer's name is not displayed. A warning may also be displayed. The possible warnings are **Unfrozen Payments** and **The Payment Event is unbalanced**.

**Payment(s)** contains the total number and value of payments linked to the payment event.

**Tender(s)** contains the total number and value of tenders linked to the payment event.

**Note.** If the payment has more than 25 payment segments, the search criteria are intentionally left blank in order to avoid retrieving all payment segments (with the resultant slow response times). You must therefore use the **obligation Filter** to define the type of payment segments that should be retrieved.

Use the **Obligation Filter** to define the types of obligations whose payment segments appear in the grid. The following options are available:

- **All.** Use this option if you do not wish to restrict payment segments based on obligation attributes.
- **Obligation Type.** Use this option to restrict payment segments to those whose obligations are linked to a given **Division** and **Obligation Type**.

Don't forget to click the search button after changing the **Obligation Filter**.

The grid that follows contains the payment segments that match your search criteria. The following information appears in the grid:

- The **Location** column contains the characteristic location associated with the payment segment's obligation. Refer to [Maintaining Locations](#) for more information about this location.
- Click the go to button to transfer to view the financial transaction related to the payment segment. This is only available after the payment has been frozen.
- **Obligation Information** contains summary information about the obligation.
- **Distributed Amt** is the amount of the obligation's debt relieved by the distribution.
- **Adjustment ID** is displayed if there is an adjustment associated with the payment.

The following information appears at the bottom of the page:

- **Payment Amount** is the amount of the payment.
- **Distributed Amount** is the sum of the payment segments linked to the payment.
- The **Difference** between the **Payment Amount** and the **Distributed Amount** appears if it is non-zero.
- **Billed Amount** is the amount of the bill preceding the payment date.
- **Delinquent Amount** is the amount of the taxpayer's debt that was due on / before the prior bill's due date.
- **Current Balance** is the account's current balance.

## Payment - Manual Distribution

You can use this page to view obligations linked to the payment's account and to manually distribute a payment amongst specific obligations.

Open this page using **Financial, Payment, Manual Distribution**.

Most payments are maintained using the [Payment Event](#) page. The transaction described below is typically only used to view a payment's payment segments and to manually distribute a payment amongst an account's obligations.

**Note.** If the payment's account has more than 25 obligations, the search criteria are intentionally left blank in order to avoid retrieving all obligations (with the resultant slow response times). You must therefore use the **Obligation Filter** to define the type of obligations that should be retrieved. See the [Description of page](#) below for more information about this page's search criteria.

### Description of page

**Payment Info** contains a concatenation of important information about the payment.

**Payment ID** is the system-assigned unique identifier of the payment.

**Note.** If the payment's account has more than 25 obligations, the search criteria are intentionally left blank in order to avoid retrieving all obligations (with the resultant slow response times). You must therefore use the **obligation Filter** to define the type of obligations that should be retrieved.

Use the **Obligation Filter** to define the types of obligations that appear in the grid. The following options are available:

- **All.** Use this option if you want to view all obligations.
- **Obligation Type.** Use this option to restrict obligations to those linked to a given **Division** and **Obligation Type**.

Don't forget to click the search button after changing the **Obligation Filter**.

The grid that follows contains the obligations that match your search criteria. The following information appears in the grid:

- **Location** contains the characteristic location associated with the obligation.
- **Obligation Information** contains summary information about the obligation.
- **Distributed Amt** is the amount of the obligation's debt relieved by the payment. This field is gray after the payment is frozen.
- **Billed Amt** is the amount that was billed for the obligation on the bill preceding the payment date.
- **Delinquent Amt** is the amount of debt associated with financial transactions that appear on overdue bills.
- **Current Balance** is the amount currently owing for this obligation.

If the account being paid is an **open item** account, then an alternate grid is displayed. The alternate grid is designed to break down payable balances by individual bill and obligation. Both unpaid bill balances and new charges are displayed. After distribution, a separate match event will be created for each bill paid or partially paid.

The open item manual distribution grid will have the following columns:

- **Bill Date**
  - Shows bill date if the row contains an account's unpaid bill.
  - Shows '**New Charges**' if the row contains an account's unbilled new charges.

- Shows '**Payment Segment**' for the following conditions:
  - Row is a pay segment of a cancelled payment.
  - Row is a pay segment of an overpayment to the highest priority obligation or an excess credit obligation.
  - Row is a pay segment of a balance forward payment distribution.
- **Location Information** contains the characteristic location associated with the obligation.
- **Obligation Information** contains summary information about the obligation.
- **Distributed Amount**
  - You can enter an amount to distribute if the payment is non-frozen and non-cancelled, otherwise, this field is protected.
- **Billed Amount**
  - The amount that was billed for the obligation on the row's bill. If the row does not represent a bill amount then this will be zero.
- **Unpaid Amount**
  - This is determined by the "determine open-item bill amount" algorithm as defined on the installation option.
  - This column will contain zero if:
    - The row is linked to an open non-disputed match event with other bills' FTs on it. In this case, it is not possible to determine the bill amount.
    - The row is linked to an open disputed match event.

For either grid, the following information appears at the bottom of the page:

- **Payment Amount** is the amount of the payment.
- **Distributed Amount** is the sum of the payment segments linked to the payment.
- The **Difference** between the **Payment Amount** and the **Distributed Amount** appears if it is non-zero.
- **Billed Amount** is the amount of the bill preceding the payment date.
- **Delinquent Amount** is the amount of the taxpayer's debt that was due on / before the prior bill's due date.
- **Current Balance** is the account's current balance.

## Payment - Characteristics

To update payment characteristics, open **Financial, Payment** and navigate to the **Characteristics** tab.

### Description of Page

The **Characteristics** collection contains information that describes miscellaneous information about the payment. The following fields display:

<b>Characteristic Type</b>	Indicate the type of characteristic.
----------------------------	--------------------------------------

<b>Sequence</b>	Controls the order in which characteristics of the same type are displayed.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

## Payment Action Codes

The topics that follow describe each of the actions that appear on the payment pages.

### Contents

- [Distribute \(A Payment\)](#)
- [Redistribute \(A Payment\)](#)
- [Freeze \(A Payment\)](#)
- [Cancel \(A Payment\)](#)
- [Transfer \(A Payment\)](#)
- [Delete \(A Payment\)](#)

### Distribute (A Payment)

Clicking the **Distribute** button causes a payment to be distributed amongst the account's obligations. Refer to [Distributing A Payment Amongst An Account's obligations](#) for a description of how this is achieved.

This button is enabled if a payment is **Incomplete**, **Error**, or **Freezable**. Refer to [Payment Lifecycle](#) for more information about these status values.

If the payment is distributed appropriately, click the [Freeze](#) button to freeze the payment and its payment segments.

### Redistribute (A Payment)

Clicking the **Redistribute** button causes a payment to be redistributed amongst the account's obligations. Refer to [Distributing A Payment Amongst An Account's Obligations](#) for a description of how this is achieved.

This button is enabled if a payment is **Incomplete**, **Error**, or **Freezable**. Refer to [Payment Lifecycle](#) for more information about these status values.

If the payment is distributed appropriately, click the [Freeze](#) button to freeze the payment and its payment segments.

**Note.** This button is only available on the [Payment - Main](#) and [Payment - Manual Distribution](#) pages.

### Freeze (A Payment)

Clicking the **Freeze** button causes a payment and its payment segments to become **frozen**. Refer to [Payment Lifecycle](#) for more information about freezing.

This button is enabled if the payment is **Freezable**.

**If problems are detected after freezing.** A payment may not be changed after it is **frozen**. All subsequent changes must occur by canceling the frozen payment and creating a new one.

## Cancel (A Payment)

Clicking the **Cancel** button causes the payment and its payment segments to be **canceled**. Canceling a payment (as opposed to canceling a tender) is typically only done if the original distribution was frozen and it's incorrect.

This button is enabled if the payment is **Frozen**. Refer to [Payment Lifecycle](#) for more information about these status values.

## Transfer (A Payment)

The **Transfer** button is enabled if the payment is **Frozen**. Refer to [Payment Lifecycle](#) for more information about these status values.

If the payment being transferred was created using payment event distribution rule(s), the payment transfer should also be done using the distribution rules method. Hence, a different dialog opens when you click **Transfer** based on whether the payment's payment event is associated with at least one distribution detail or not.

### Contents

[Transfer Payment \(No Distribution Details Exist\)](#)

[Transfer Payment \(Distribution Details Exist\)](#)

#### Transfer Payment (No Distribution Details Exist)

Payment transfer causes the payment and its payment segments to be **canceled** and a new payment to be created. Transferring a payment is typically only done if the original distribution was targeted at the wrong account but all tender information for the pay event is correct. Unlike the [payment event transfer](#), transferring a single payment does not affect the pay event's tender information.

You must specify the following parameters in order to transfer a payment:

- **Account ID** is the account to which the payment should be transferred.
- **Cancel Reason** defines why you are performing the transfer.
- **Match Type** and **Match Value** are defaulted based on their values for the payment event that you are transferring. If necessary, modify the values appropriately for the account to which you are transferring the payment event. Match Type and Match Value are used for an account that belongs to an [open item](#) account type or to restrict the distribution of the payment to a specific obligation. Refer to [Payment Event - Main](#) for more information.
- Turn on **Freeze Payment** if the transferred payment should be frozen automatically. You may want to leave this turned off if you want to examine the payment distribution before freezing the payment.
- **Non Tax Agency Name**, **Reference Number**, and **Non Tax Agency Comments** appear if the **Account ID** belongs to an account type that is used for non tax agency payments. Refer to [Setting Up Account Types](#) for more information.

Clicking **OK** causes the following to take place:

- The original payment and its segments are cancelled.
- A new payment is added for the transfer to account. The payment characteristics associated with the original payment are copied to the new payment.
- The payment is distributed amongst the new account's obligations.

- If so designated, the new payment is frozen.

#### Transfer Payment (Distribution Details Exist)

Payment transfer causes the payment to be **canceled** and new payment(s) to be created using new payment event distribution rule(s). Transferring a payment is typically done when a payment event has multiple payments and one or more payments have been assigned to the wrong obligation. Each of the incorrectly directed payments can be transferred to the correct obligation(s) using this dialog.

**Total Amount** is the total payment amount to transfer from.

You must specify the following parameters in order to transfer a payment:

- **Distribution Rule** is the obligation to which the payment should be transferred.
- **Rule Value** is the value associated with the payment and expected by the distribution rule.
- **Amount** is the payment amount.
- **Cancel Reason** defines why you are performing the transfer.

Clicking **OK** causes the following to take place:

- The payment associated with the original account is canceled.
- The new set of distribution details is processed, creating new payment(s) for the transfer-to obligation(s). The payment characteristics associated with the original payment are copied to the new payment.
- The new set of distribution details is added to the payment event. The original distribution details are kept.

**Distribution rule entries can have a zero amount.** Refer to [Rule Value Can Capture Additional Information](#) for more information about how to use distribution details to capture additional payment related information.

**Determine Tender Account.** This dialog does not call the **Determine Tender Account** algorithm defined on [distribution rule](#) as this action does not rebuild the payment tender collection.

### Delete (A Payment)

The **Delete** button deletes a payment. This button is enabled if the payment is **Incomplete**, **Error**, or **Freezable**.

## How To

---

The topics in this section describe how to perform common payment maintenance functions.

#### Contents

[How To Add A New Payment Event](#)  
[How To Cash A Check](#)



[How To Allocate The Tender Amount To Multiple Accounts](#)  
[How To Print Receipts And Endorsements](#)  
[How To Cancel A Tender](#)  
[How To Transfer A Payment From One Account To Another](#)  
[How To Distribute A Payment To A Specific obligation](#)  
[How To Get An Unbalanced Tender Control In Balance \(Fixing Over/Under\)](#)

## How To Add A New Payment Event

There are several ways to add a new event:

- You can use [Payment QuickAdd](#) to quickly add one or more payment events. You'd use this approach to add simple payments where no manual intervention is required. By "simple payment" we mean:
  - The account is both the tendering account and the account whose debt is being relieved by the payment
  - The payment date is the current date
  - The payment should be [distributed](#) amongst the account's obligations using standard distribution logic
- If applicable to your business practice, you can use [Payment Event QuickAdd](#) to quickly add one or more payment events using [distribution rules](#).
- You can use [Payment Event Maintenance](#) to add a payment event. You would use this approach if multiple forms of payment are remitted (e.g., cash and a check) or if there are multiple payors and/or payees linked to the payment event.

**Note.** By default the system opens the [Payment Event - Add Dialog](#) when you navigate to this page in add mode. If you have opted to always use the payment event distribution rules method as your [default method](#), the [Payment Event QuickAdd \(Single Payment Event\)](#) page appears instead. Refer to these pages for more information.

## How To Cash A Check

If you allow taxpayers to cash checks, you'll have a payment event with two tenders:

- The first tender contains information about the check.
- The second tender contains information about the cash refund (the tender amount is a negative amount equal to the amount of the check).

The interesting aspect of this payment event is that it has no payments because the sum of the tenders is zero.

We start this explanation in the middle of [How To Add A New Payment Event](#).

In the **Payment Event Add** dialog, be sure to select **Do Not Distribute** before clicking **OK**.

- On the **Main** page, click – to remove the payment (cashing a check is a payment event with no payments).
- Transfer to the **Payment Event, Tenders** page.

- **Insert** a row in the tender scroll (click the + button) for the cash refund tender. In this row enter a tender type of “cash” and a tender amount of negative the check amount.
- Save the event (note there is nothing to distribute or freeze).

## How To Allocate The Tender Amount To Multiple Accounts

When you add a payment event, the system automatically creates a single payment for the account that remitted the funds. If someone is remitting funds for someone other than themselves, you must change/add payments. This section describes how to do this.

**Note.** This section assumes you chose to add the payment event using the [Payment Event - Add Dialog](#). Refer to [How To Add A New Payment Event](#) for the complete list of options.

In the [Payment Event - Add Dialog](#), be sure to select **Do Not Distribute** before clicking **OK**.

Once on the payment event page, go to the Tenders tab and do the following:

- If the remitting account shouldn't have received any part of the payment, **Remove** it by clicking the - button. Alternatively, you can just change the account id to reflect the recipient.
- If multiple accounts receive the remitted funds, **Insert** one row in the payment scroll (click the + button) for each additional account.
- When the payment event is balanced, **Save** it. Then **Distribute** and **Freeze** the payment(s). Refer to [Payment Event Actions](#) for more information on these action buttons.

## How To Print Receipts And Endorsements

The system can be configured to allow you to endorse checks and print receipts using special cashiering station printers. This section describes the print options available.

**Note.** These options are only available if they have been installed in your system. Installing these options are a delivery and installation issue and are not within the domain of this document.

The print functions are available from the [Payment Event](#), [Payment QuickAdd](#) and [Payment Event QuickAdd](#) transactions. Refer to those sections for more information.

### Description of Page

Use the **Endorse** button to endorse checks using the cashier station printer. This is only enabled if one of the tender types indicates a check. If there were multiple checks, use the scroll buttons to select them. Feed the appropriate check in the printer and an endorsement message is printed on the back.

Use the **Receipt** button to print a receipt using the cashier station printer. Choose a **Short** or **Long** option. Use the **Duplicate** button to print a duplicate receipt – this prints the receipt with a “duplicate” label.

Use the **Stub** button to print special information in bill stub format (account, name, amount). Feed a slip of paper into the printer and the special stub information will be printed.

When you are finished, click **Done**; click **Cancel** to exit at any time.


The various text strings that are printed on the receipt and the endorsement are defined on [Installation Options - Messages](#).

## How To Cancel A Tender

If a tender is no longer valid (e.g., a check bounces), the tender must be canceled.

**Warning!** This process only makes sense in the context of the page used to maintain tenders. Refer to [Payment Event - Tenders](#) for the details.

You navigate through the pages as follows:

1. Open the payment event - tenders page for the tender in question. Proceed to step 2.
2. Click . Proceed to step 3.
3. Before the system cancels the tender, you must specify the cancel reason.

Select a **Cancel Reason** to describe why the tender is being canceled. The system also needs to know which **Bank Account To Charge** the cancellation against. The system will default the **Bank Code** and **Bank Account** from the original bank information used when the tender was deposited. You can override them here.

When you click the **OK** button, the system cancels the tender and ALL payments linked to the event.

If the cancellation reason you supply is indicated as being "non-sufficient funds", the system will generate an adjustment to levy a NSF charge. Refer to [Obligation Type - Main Information](#) for more information.

**When non-canceled tenders are still linked to the event.** If there are multiple tenders linked to the payment event, you must either add new payment(s) that equal the sum of the event's non-canceled tenders or cancel the remaining tenders.

**Cancellation after cash back.** If a taxpayer tenders a check for \$100, but only owes \$80, the system will recommend returning \$20 in cash to the taxpayer (assuming the tender type for checks allows overpayment). If the tender type for checks is not "like cash", a second tender is created for -\$20 (the first tender is for \$100). If the check subsequently bounces, both tenders must be cancelled.

## How To Transfer A Payment From One Account To Another

If you need to transfer a payment amount from one account to another, you can use the **Transfer** button either on the payment event page or the payment page.

Refer to [Transferring A Payment](#) for a description of when to use each option.

## How To Distribute A Payment To A Specific obligation

When you click **Distribute**, the system uses the payment distribution algorithm defined on the account's account type to allocate the payment amongst the account's obligations. In this section, we describe how to override this distribution (e.g., when you need to allocate a payment to specific obligation(s)).

We start this explanation in the middle of [How To Add A New Payment Event](#).

In the **Payment Event Add** dialog, be sure to select **Do Not Distribute** before clicking **OK**.

- **Save** the payment event (this also saves the payment and tender information).
- Use the **Payment Amount** context menu to transfer to the **Manual Distribution** page.
- Fill in the desired **Distributed Amt** for each obligation and save the payment.
- Click **Freeze** to freeze the payment when the amount distributed is equal to the payment amount.

## How To Get An Unbalanced Tender Control In Balance (Fixing Over/Under)

In order to balance a tender control that is out-of-balance, you must create a tender for an account associated with your company.

**The over / under account.** Your organization must set up a company-use account with an obligation whose obligation type references the over/under distribution code. This account should be linked to the person ID associated with company usage.

- If you have more money in the drawer than you have tenders, then you're "over". In this situation, you need to record a payment for the "over" amount. This will cause cash to be debited by the over amount, and the expense account associated with the account's obligation to be credited.
- If you have less money in the drawer than you have tenders, then you're "under". In this situation, you need to record a negative payment for the "under" amount. This will cause cash to be credited by the over amount, and the expense account associated with the account's obligation to be debited.

**Important!** You must determine the over/under amount for each tender type and then enter the respective over/under tender for the company-use account. For example, if you have more cash but fewer checks then you must enter two tenders for the company-use account (but you can do this on the same event). Check out the following table for an example:

Tender Type	Starting Balance	Tenders Received	Turn Ins	Actual Ending Balance (Entered by	Expected Ending Balance	Over/Under Amount

				Cashier)		
Cash	\$150.50	\$5,000	\$4,000	\$1151	\$1150.50	Over \$0.50
Check	-	\$1,000	\$750	\$249	\$250	Under \$1

In this example, you'd have to create two tenders for the company-use account:

- Tender type: Cash, Tender amount: +\$0.50
- Tender type: Check, Tender amount: -\$1.00

The sum of the tenders is -\$0.50 and this is the amount that will be distributed to the company-use account's obligation (debit over/under expense, credit cash).

**You must reopen the tender control.** Before you can add an over/under tender to a tender control, you must **reopen** the tender control (tenders may only be added to **open** tender controls).

## Financial Transactions On A Payment

Open **Financial Query, Financial Transactions On A Payment** to view the financial transactions on a payment.

**Note.** You can also open this page using the go to buttons that prefix the financial transaction summaries on [Payment - Main](#).

### Description of page

**Payment Id** is the system-assigned unique identifier of the payment.

**Account ID** is the payment's account.

The area beneath **Account ID** provides you with options that control which financial transactions appear in the grid. The following points describe the various options:

- Use the **Obligation Filter** to define the types of obligations whose financial transactions appear in the grid. The following options are available:
  - **All.** Use this option if you do not wish to restrict financial transactions based on obligation attributes.
  - **Obligation ID.** Use this option to restrict financial transactions to those of a specific obligation.
  - **Obligation Type.** Use this option to restrict financial transactions to those whose obligations are linked to a given **Division** and **Obligation Type**.
- Use **Match Event Status Filter** to restrict the financial transactions based on the status of their [match event](#). This filter only appears if the payment's account is an [open-item](#) taxpayer. The following options are available:
  - **All.** This option shows all financial transactions.
  - **Balanced.** This option shows all financial transactions whose match event is **balanced**.

- **Disputed.** This option shows all financial transactions whose match event is **disputed**.
- **Unbalanced.** This option shows all financial transactions whose match event is **unbalanced**.
- **Unmatched.** This option shows all financial transactions that are not linked to a match event.

Don't forget to click the search button after changing the filters or after selecting a new Payment Id.

The grid that follows contains the financial transactions that match your search criteria. The following information is displayed:

- **Match Event Status** shows the status of the financial transaction's [match event](#). This column only appears if the account is an [open-item](#) taxpayer.
- **FT Type** displays the type of financial transaction. Click on the hyperlink to transfer to [Financial Transaction - Main](#). On this page, you can change certain aspects of the FT in question.
- **Accounting Date** is the date the system uses to determine the financial transaction's accounting period in your general ledger.
- **Current Amount** contains the financial transaction's effect on the obligation's current balance.
- **Payoff Amount** contains the financial transaction's effect on the obligation's payoff balance. The **Payoff Amount** will be dim if it equals the **Current Amount**.
- **Show on Bill** indicates if information about the financial transaction appears on the taxpayer's bill.
- **Obligation Information** contains a summary of the respective obligation.
- **Financial Transaction ID** is the system-assigned unique identifier of the financial transaction.

At the bottom of the page is a summary of the financial transactions that match the search criteria.

## Payment History

The payment history page shows all payments that have been distributed to an account's obligations. Open this page using **Financial Query, Account Payment History**.

### Description of Page

One row is displayed for every payment that has been distributed to an account's obligations. The payment date, amount, payment status and the tender source associated with the tender's tender control are displayed for each payment.

**Note.** **Tender Source** typically contains the description of the cash drawer in which the payment was made or the remittance processor that processed the payment. **Tender Source** will be blank for automatic payments until they are interfaced to the financial institution. Refer to [Downloading Automatic Payments](#) for more information about interfacing automatic payments.

If you need to see more detailed information about the payment, click the Go To button to transfer to the payment event page.

## Payment / Tender Search

This page allows you to look for payments and/or tenders using a combination of search criteria. Open this page using **Financial Query, Payment / Tender Search**.

### Description of Page

The top half of the page is where you enter the criteria used to search for payments and tenders.

**Multiple search criteria may be specified.** You can search for payments and tenders using a combination of search criteria. For example, if you enter both a **Payment Account** name of **Brandon** and a **Payment Amount** between **150** and **170**, only those payments for taxpayers named **Brandon** whose amount is between **150** and **170** will be displayed.

**Warning!** Try to be as specific as possible when entering search criteria. Why? Because entering open-ended search criteria may have a severe impact on response times. For example, if you know the payments you're looking for have a payment date sometime in **January 2003** and the taxpayer's name is **Brazil,John**, then enter both of these criteria. If you also know the payment amount is between **150** and **170**, enter these values too.

The following table describes each of the different search methods.

Search Method	Description
Search for	Use this field to define if you're searching for <i>Payments</i> , <i>Tenders</i> or both <i>Payments and Tenders</i> . The value entered in this field controls which of the remaining search methods are enabled.
Distribution Rule	Use this field if you're searching for a tender or a payment and know the distribution rule and value used to created it. <b>Note.</b> This section appears only if you have <a href="#">configured</a> your system to allow the payment event distribution rules method to be used.
Payment Account	Use this field if you're searching for a payment and know the account ID or the name of any person linked to the account whose debt is relieved by the payment. - If you know the payment's account ID, first choose <i>Account</i> in the adjacent dropdown and then enter the account's ID. You must enter all of the account ID. - If you know the name of any person linked to the account, first choose <i>Person Name</i> in the adjacent dropdown and then enter the person's name. You can enter all or part of the person's name (the more you enter, the faster the search will be). The name search is not case sensitive. A search method of <i>Person Name</i> defaults. If you leave the person name or ID blank, the system ignores this search method. These fields are protected if you've selected a <b>Search for</b> of <i>Tender</i> .
Payment Amount	Use this field if you're searching for a payment and know its amount.

Search Method	Description
	<p>- If you know the exact amount, first choose <i>Equal To</i> in the adjacent dropdown and then enter the amount.</p> <p>- If you know the payment amount is between a given range of values, first choose <i>Between</i> and then enter the range of values.</p> <p>A search method of <i>Between</i> defaults.</p> <p>If you leave the amount(s) blank, the system ignores this search method.</p> <p>These fields are protected if you've selected a <b>Search for</b> of <i>Tender</i>.</p>
Payor Account	<p>Use this field if you're searching for a tender and know the account ID or the name of any person linked to the tendering account.</p> <p>- If you know the tender's account ID, first choose <i>Account</i> in the adjacent dropdown and then enter the account's ID. You must enter all of the account ID.</p> <p>- If you know the name of any person linked to the tendering account, first choose <i>Person Name</i> in the adjacent dropdown and then enter the person's name. You can enter all or part of the person's name (the more you enter, the faster the search will be). The name search is not case sensitive.</p> <p>A search method of <i>Person Name</i> defaults.</p> <p>If you leave the person name or ID blank, the system ignores this search method.</p> <p>These fields are protected if you've selected a <b>Search for</b> of <i>Payment</i>.</p>
Tender Amount	<p>Use this field if you're searching for a tender and know its amount.</p> <p>- If you know the exact amount, first choose <i>Equal To</i> in the adjacent dropdown and then enter the amount.</p> <p>- If you know the tender amount is between a given range of values, first choose <i>Between</i> and then enter the range of values.</p> <p>A search method of <i>Between</i> defaults.</p> <p>If you leave the amount(s) blank, the system ignores this search method.</p> <p>These fields are protected if you've selected a <b>Search for</b> of <i>Payment</i>.</p>
Tender Source	<p>If you're searching for a tender and you know the source of the tender (e.g., the lockbox, cashier, etc.), enter the tender source code.</p> <p>A search method of <i>Equal To</i> defaults.</p> <p>If you leave the tender source blank, the system ignores this search method.</p> <p>These fields are protected if you've selected a <b>Search for</b> of <i>Payment</i>.</p>
MICR ID	<p>Use this field if you're searching for a tender and know its MICR (magnetic ink character recognition) ID.</p> <p>- If you know the entire MICR ID, first choose <i>Equal To</i> in the adjacent dropdown and then enter the entire MICR ID.</p> <p>- If you know the first X characters of the MICR ID, first choose <i>Like</i> in the adjacent dropdown and then enter the entire MICR ID.</p> <p>A search method of <i>Like</i> defaults.</p> <p>If you leave the MICR ID blank, the system ignores this search method.</p> <p>These fields are protected if you've selected a <b>Search for</b> of <i>Payment</i>.</p>



Search Method	Description
Payment Date	<p>Use this field if you're searching for a tender or a payment and know its date.</p> <ul style="list-style-type: none"> <li>- If you know the exact date, first choose <a href="#">Equal To</a> in the adjacent dropdown and then enter the date.</li> <li>- If you know the date is between a given range of values, first choose <a href="#">Between</a> and then enter the date range.</li> </ul> <p>A search method of <a href="#">Between</a> defaults.</p> <p>If you leave the date(s) blank, the system ignores this search method.</p>

The system shows the total number of payments / tenders that satisfy your search results immediately below the grid.

The first group of payments / tenders is displayed in the grid at the bottom of the page. Different columns appear in the grid depending on the value of **Search for**.

- If you use a **Search for** of [Payment](#), only payment-oriented columns appear in the grid.
- If you use a **Search for** of [Tender](#), only tender-oriented columns appear in the grid.
- If you use a **Search for** of [Payment and Tender](#), both payment-oriented and tender-oriented columns appear in the grid.

**Warning!** If you use the [Payment and Tender](#) search method, and the resultant tenders / payments are linked to payment events that have multiple payments or tenders, multiple rows may be displayed for the payment event's tenders and payments. For example, if a tender with multiple payments is selected, a separate row will be displayed for every payment that matches your payment search criteria. If you don't specify any payment search criteria, a row will be displayed for every payment linked to the tender.

<b>Payment Date</b>	This contains the date of the payment event associated with the payment / tender. This column appears regardless of the value of <b>Search for</b> .
<b>Payment Account Info</b>	This contains a concatenation of important information about the account whose debt was relieved by the payment. This column does not appear if you use a <b>Search for</b> of <a href="#">Tender</a> .
<b>Payment Amount</b>	This contains the amount of the payment. This column does not appear if you use a <b>Search for</b> of <a href="#">Tender</a> .
<b>Payment Status</b>	This contains the status of the payment. Refer to <a href="#">Payment Lifecycle</a> for more information. This column does not appear if you use a <b>Search for</b> of <a href="#">Tender</a> .
<b>Payment ID</b>	This contains the unique identifier of the payment. This column does not appear if you use a <b>Search for</b> of <a href="#">Tender</a> .

<b>Payor Account Info</b>	This contains a concatenation of important information about the account who tendered the payment. This column does not appear if you use a <b>Search for</b> of <a href="#">Payment</a> .
<b>Tender Amount</b>	This contains the amount of the tender. This column does not appear if you use a <b>Search for</b> of <a href="#">Payment</a> .
<b>Tender Control ID</b>	This contains the related tender control. This column does not appear if you use a <b>Search for</b> of <a href="#">Payment</a> .
<b>MICR ID</b>	This contains the MICR (magnetic ink character recognition) ID of the tender. This column does not appear if you use a <b>Search for</b> of <a href="#">Payment</a> .
<b>Tender Status</b>	This contains the status of the payment. Refer to <a href="#">Tender Lifecycle</a> for more information. This column does not appear if you use a <b>Search for</b> of <a href="#">Payment</a> .
<b>Pay Tender ID</b>	This contains the unique identifier of the tender. This column does not appear if you use a <b>Search for</b> of <a href="#">Payment</a> .
<b>Pay Event ID</b>	This contains the unique identifier of the payment event.

## Payment Event Exception

Unbalanced payment events cause a record to be written to the payment event exception table with a message indicating the nature of the error.

To view the messages associated with the exception records, schedule the [TD-UNBAL](#) background process. This process generates a To Do entry for every record in the payment event exception table.

Refer to [Unbalanced Payment Events](#) for instructions describing how to correct an unbalanced payment event.

## Payment Exception

When the system is unable to distribute a payment, a record to be written to the payment exception table with a message indicating the nature of the error.

To view the messages associated with the exception records, schedule the [TD-PYERR](#) background process. This process generates a To Do entry for every record in the payment exception table.

After correcting the cause of the error, drill into the [Payment](#) and attempt to redistribute it.

## Maintaining Deposit Controls

Deposit controls exist to give you administrative control over your cash drawers (and all other tender sources) and the subsequent deposit of funds at banks.

Refer to [Tender Management and Workstation Cashiering](#) for background information.

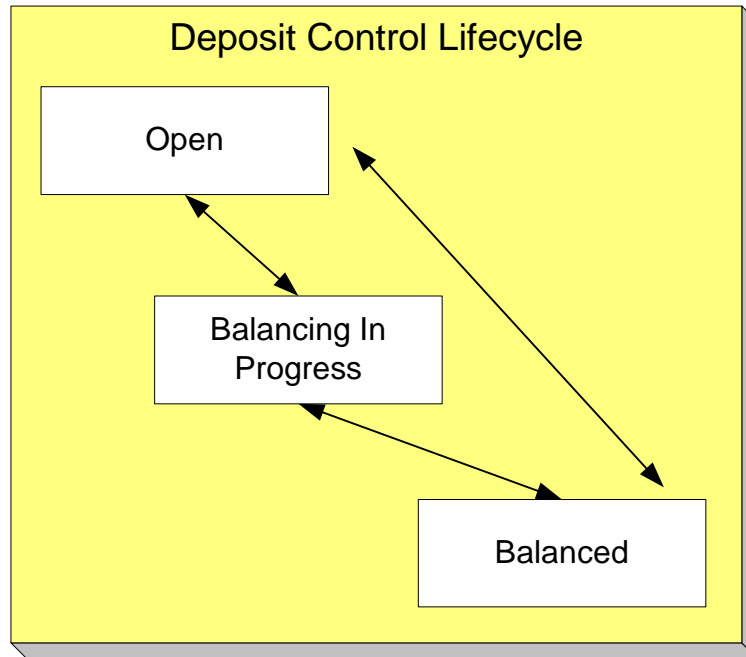
**The system creates most deposit controls behind-the-scenes.** Most deposit controls are created by the system when it processes tenders from your remittance processor and lock boxes. You should only have to access the deposit control pages if you record payments in cash drawers. For information about how the system creates deposit controls, refer to [Managing Payments Interfaced From External Sources](#). Also note that the automatic payment activation process also creates tender and deposit controls. Refer to [Activating Automatic Payments](#) for more information.

### Contents

- [The Lifecycle Of A Deposit Control](#)
- [Deposit Control - Main](#)
- [Deposit Control - Tender Control](#)
- [Deposit Control - Tender Deposit](#)
- [Deposit Control - Turn Ins](#)

## The Lifecycle Of A Deposit Control

The following diagram shows the possible lifecycle of a deposit control.

**Open**

A deposit control is initially created in the **Open** state. While in this state, you may add new deposits to it, change the deposit amount on its deposits, and transfer tender controls into and out of it.

**Balancing In Progress**

You change a deposit control's status to **Balancing In Progress** when you're ready to balance its contents. While in this state, you can change the deposit amount on its deposits and transfer tender controls out of it. If you need to add new deposits to it or transfer tender controls into it you must return it to the **Open** state.

**Balanced**

You change a deposit control's status to **Balanced** when the sum of its tender controls is consistent with the total of its deposits. While in this state, you cannot modify its deposits or its tender controls. If you need to make modifications, you must return it to the **Open** or **Balancing In Progress** state.

**Background processes and state transition.** When payments are interfaced from external sources, the system automatically creates a deposit control and links a tender control to it. When all payments have been successfully loaded, the system changes the state of the respective deposit control to **Balanced**.

## Deposit Control - Main

The Main page contains core deposit control information. Open this page using **Financial, Deposit Control** and then navigate to the **Main** tab.

**Description of Page**

**Deposit Control** contains a concatenation of the deposit control's creation date, tender source type, and status.

**Deposit Control ID** is the system-assigned, unique identifier of the deposit control.

**User** is the person who created the deposit control.

**Create Date/Time** is the date and time the deposit control was created.

**Tender Source Type** is the type of tender control that has been linked to the deposit control. Valid values are: *Ad hoc*, *Auto Pay*, *Online Cashiering* and *Lockbox*. The system uses this information to prevent tender controls from different sources from being included under the same deposit control. In other words, you can't mix automatic payment, cashiering and lockbox tenders under the same deposit control.

**Currency Code** is the currency in which the deposit control's tenders are denominated.

**Default note.** The currency code defaults from the installation record.

The summary information that follows contains a summary of the starting balance and the tenders that are linked to the tender control.

- **Starting Balance** is the sum of the starting balances from all tender controls linked to the deposit control.
- **Total Tenders Amount** is the sum of tenders from all tender controls linked to the deposit control.
- **Total Tender Controls** is the number and amount of tender controls linked to the deposit control.
- **Total Tender Deposits** is the number and amount of tender deposits linked to the deposit control.
- **Expected Ending Balance** is the **Total Tender Control** minus **Total Tender Deposits**.
- **Ending Balance** is the actual amount of money in the tender control. This amount must equal the **Expected Ending Balance** before the tender control can be marked as *Balanced*.
- **Outstanding Over/Under** is **Ending Balance** minus **Expected Ending Balance**.

**Deposit Control Status** shows the status of the deposit control. Valid values are *Open*, *Balanced*, and *Balancing In Progress*.

For more information, refer to [The Lifecycle Of A Deposit Control](#).

The **Balanced User ID** and **Balanced Date/Time** are populated when the status is changed to *Balanced*.

Use **Comments** to describing anything unusual about the deposit control.

## Deposit Control - Tender Control

The Tender Control page contains a row for every tender control linked to the deposit control. Open this page using **Financial, Deposit Control, Tender Control**.

**Description of Page**

**Deposit Control** contains a concatenation of the deposit control's creation date, tender source type, and status.

**Deposit Control ID** is the system-assigned, unique identifier of the deposit control.

The grid that follows contains a row for every tender control linked to the deposit control. No information about the tender controls may be modified on this page. To view or modify tender control information, click the Go To button.

## Deposit Control - Tender Deposit

The Tender Deposit tab page contains a row for every tender deposit linked to the deposit control. Open this page using **Financial, Deposit Control** and navigate to the **Tender Deposit** tab.

**Description of Page**

**Deposit Control** contains a concatenation of the deposit control's creation date, tender source type, and status.

**Deposit Control ID** is the system-assigned, unique identifier of the deposit control.

The **Tender Deposit** scroll that follows contains one row for every financial institution into which the tenders will be deposited. To insert a new row, click the + button and fill in the following fields:

- **Tender Deposit ID** is maintained by the system. When a deposit is being created, there is no ID number displayed. Once a deposit is entered and saved, the system generates an ID and displays it here.
- **Deposit Amount** contains the amount to be deposited at the bank. **Currency Code** is the currency in which the deposit is denominated.
- **Reference ID** contains the ID of the deposit (if any).
- Use **Bank Code** and **Bank Account** to define where the tenders will be deposited.

## Deposit Control - Turn Ins

The Turn Ins page contains a row for every turn-in event linked to the deposit control. Open this page using **Financial, Deposit Control** and navigate to the **Turn Ins** tab.

Refer to [Turn Ins](#) for background information.

**Description of Page**

**Deposit Control** contains a concatenation of the deposit control's creation date, tender source type, and status.

**Deposit Control ID** is the system-assigned, unique identifier of the deposit control.

The turn-ins grid contains one row for every turn-in event associated with the deposit control. A turn-in event is created when moneys originally deposited in respect of a tender control are turned in to the tender control's deposit control. Turn-in events are created and maintained on the [Tender Control – Turn Ins](#) page. Refer to [Turn Ins](#) for background information. The following information is displayed in the grid:

- **Turn In Status** defines if it is a new turn in *Awaiting approval* or has been *Approved* by the operator who is responsible for the Deposit Control. Once the turn-in is *Approved*, this field becomes display-only.
- **Tender Type** is the type of tender that has been turned in. This is a display-only field.
- **Turn In Amount** is the amount of the **Tender Type** that has been turned in. This is a display-only field.
- **Receipt Number** is the ID of the receipt given to the person who turn-in the funds. This is a display-only field.
- **Create Date/Time** contains when the turn-in event was created. This is a display-only field.

## Maintaining Tender Controls

Tender controls exist to give you administrative control over your cash drawers (and all other tender sources).

Refer to [Managing Your Cash Drawers](#) for more information.

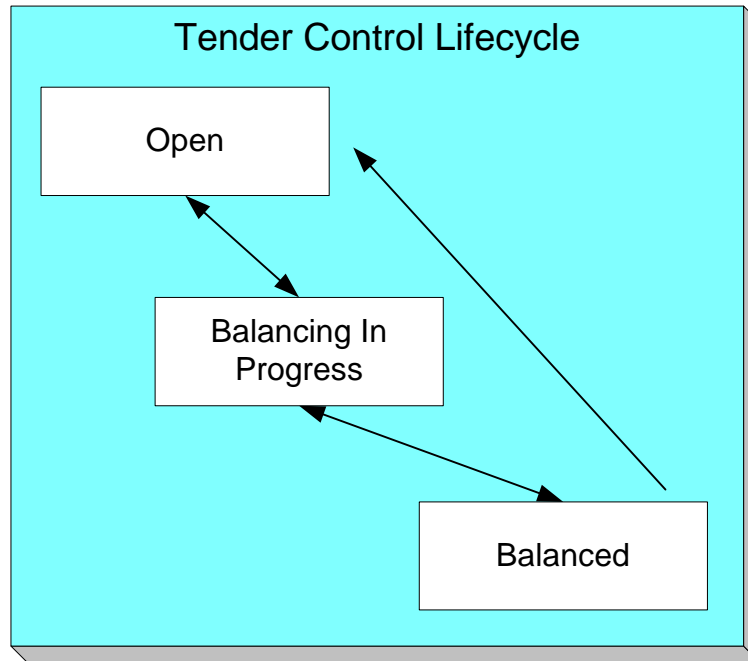
**The system creates most tender controls behind-the-scenes.** Most tender controls are created by the system when it processes tenders from the remittance processors and lock boxes. You should only have to access the tender control pages if you record payments in cash drawers. For information about how the system creates tender controls, refer to [Managing Payments Interfaced From External Sources](#). Also note that the automatic payment activation process also creates tender and deposit controls. Refer to [Activating Automatic Payments](#) for more information.

### Contents

- [The Lifecycle Of A Tender Control](#)
- [Tender Control - Main](#)
- [Tender Control - Tenders](#)
- [Tender Control - Turn Ins](#)
- [Tender Control - Exceptions](#)
- [Tender Control - Characteristics](#)

## The Lifecycle Of A Tender Control

The following diagram shows the possible lifecycle of a tender control.

**Open**

A tender control is initially created in the **Open** state. While in this state, you may add new tenders to it, change the tender amount on its tenders, and transfer tenders into and out of it.

**Balancing In Progress**

You change a tender control's status to **Balancing In Progress** when you're ready to balance its contents. While in this state, you can change the tender amount on its tenders and transfer tenders out of it. If you need to add new tenders to it or transfer tenders into it, you must return it to the **Open** state.

**Balanced**

You change a tender control's status to **Balanced** when the sum of its tenders is consistent with the ending balance in the drawer. While in this state, you cannot modify it or its tenders. If you need to make modifications, you must return it to the **Open** or **Balancing In Progress** state.

If the tender control is part of a **Balanced** deposit control, you may not change its status.

**Background processes and state transition.** When payments are interfaced from external sources, the system automatically creates a tender control and links tenders to it (one for each payment interfaced). When all payments have been successfully loaded, the system changes the state of the respective tender control to **Balanced**.



## Tender Control - Main

The Main page contains core tender control information. This page is used to balance the tender control.

Open this page using **Financial, Tender Control**.

**Searching For Tender Controls.** When you use the **Creation Date** to search for tender controls, the system returns tender controls that are created on or before the date you specify. If the number of records returned exceeds the search limit (i.e. 300 records), only the records that do not exceed search limit are displayed. Therefore, you may need to specify an earlier creation date to find the record you are looking for. If you do not specify a creation date and search based on other criteria, the system uses the most recent date.

Refer to [Managing Your Cash Drawers](#), [Turn Ins](#) and [Balancing By Tender Type](#) for more information.

### Description of Page

**Tender Control** contains a concatenation of the tender control's creation date, tender source, and status.

**Tender Control ID** is the system-assigned, unique identifier of the tender control.

**Tender Source** is the source of the tenders (e.g., cash drawer 22, lockbox 1 at Bank of America).

**Deposit Control ID** is the system-assigned, unique identifier of the deposit control that the tender control is part of. The deposit control's **Currency Code** is displayed below.

For more information about deposit controls and tender controls, refer to [Managing Your Cash Drawers](#).

Turn on **All Users** if any user may insert tenders into this tender control. Turn this switch off and specify the appropriate **User ID** if only a single user may insert tenders into this tender control.

**Create Date/Time** is the date and time the tender control was created.

The **Starting Balance** of the tender control is defaulted from the tender source. This may be overridden while the **Tender Control Status** is **Open** or **Balancing in Progress**.

The **Tender Control Status** shows the status of the tender control. The **Balanced User ID** and **Balanced Date/Time** are populated when the status is changed to **Balanced**.

For more information, refer to [The Lifecycle Of A Tender Control](#).

The **Balance** button is enabled when the **Tender Control Status** is **Balancing in Progress**. When this button is clicked, the system checks the following:

- All Turn-Ins have been **Approved**.
- The sum of all **Ending Balances** equals the sum of all **Expected Ending Balances**.

If the above conditions are true, the status of the tender control is changed to **Balanced** and all modifiable fields become protected.

The tenders grid contains a summary of the tenders linked to the tender control. One row is displayed for each tender type. Each row contains the number of tenders of this type and their total amount. In order to **Balance** the tender control, you must enter an appropriate **Ending Balance** for each **Tender Type**. Refer to [Balancing By Tender Type](#) for more information. The following information appears in the grid:

- **Tender Type** represents the type of tender (e.g., cash, credit card). This is a display-only field.
- **Number of Tenders** contains the total number of tenders of this type in the tender control. This is a display-only field and is calculated by the system by accumulating the tenders linked to the tender control.
- **Tender Total Amount** contains the total amount of tenders of this type in the tender control. This is a display-only field and is calculated by the system by accumulating the tenders linked to the tender control. Navigate to the **Tenders** page to see the individual tenders.
- **Turn In Amount** contains the total amount of turn-ins of this type. This is a display-only field and is calculated by the system by accumulating the turn-ins linked to the tender control. Navigate to the **Turn Ins** pages to see the individual turn ins.
- **Starting Balance** contains the starting balance of this type of turn in. This field is only populated on the row defined as the Starting Balance tender type on the Installation Record. This is a display-only field and is equal to the tender control's **Starting Balance**.
- **Expected Ending Bal** is the **Starting Balance** plus **Tender Total Amount** minus **Turn In Amount** for this type of tender.
- **Ending Bal** is the actual amount of money of this tender type. The cashier must enter this field in order to balance the tender control. This field is protected unless the **Status** of the tender control is **Balancing In Progress**. This amount must equal the Expected Ending Balance before the tender control can be marked as **Balanced**. Refer to [How To Get An Unbalanced Tender Control In Balance \(Fixing Over/Under\)](#) for more information.
- **Outstanding Over/Under** is the difference between **Expected Ending Bal** and **Ending Bal**. This is a display-only field.

The summary information beneath the tender type grid contains a summary of the tenders that are linked to the tender control.

- **Total Tender Amount** is the total amount of all tenders linked to the tender control.
- **Ending Balance** is the actual amount of money in the tender control (as defined in the above grid).
- **Expected Ending Balance** is the expected amount of money in the tender control (as defined in the above grid).
- **Outstanding Over/Under** is Ending Balance minus Expected Ending Balance. If this amount is zero, the field is not displayed.

If the tender controls were created for a background process, such as the automatic payment extract, the following information is displayed:

- **Batch Code** is the batch code for which the tender control was created. This field only appears if it contains a value.

- **Batch Number** is the batch number for which the tender control was created. This field only appears if it contains a value.

Use **Comments** to describe anything unusual about the tender control (e.g., to explain why a large over/under amount was created).

## Tender Control - Tenders

The Tenders page contains a row for every tender linked to the tender control. Open this page using **Financial, Tender Control, Tenders**.

### Description of Page

**Tender Control** contains a concatenation of the tender control's creation date, tender source, and status.

**Tender Control ID** is the system-assigned, unique identifier of the tender control.

This grid at the top of the page contains a row for every tender linked to the tender control. When first displayed, the grid is ordered by the **Create Date/Time** column. No information about the tenders may be modified. To view or modify tender information, click the Go To button.

The grid at the bottom of the page contains a summary by tender type of all tenders linked to the tender control.

## Tender Control - Turn Ins

Every time you turn in funds to the deposit control that is associated with your tender control, you create a turn-in event using this page. Open this page using **Financial, Tender Control** and navigate to the **Turn Ins** tab.

Refer to [Managing Your Cash Drawers](#) and [Turn Ins](#) for more information.

### Description of Page

**Tender Control** contains a concatenation of the tender control's creation date, tender source, and status.

**Tender Control ID** is the system-assigned, unique identifier of the tender control.

The turn-ins grid contains one row for every turn-in event associated with the tender control. A turn-in event is created when moneys are physically transferred to the deposit control associated with the tender control. Turn-in events are approved on the [Deposit Control – Turn Ins](#) page. The following information is displayed in the grid:

- **Turn In Status** defines if the turn-in has been **Turned In** or **Approved** by the operator who is responsible for the Deposit Control. When a turn-in is first created, its status is **Turned In** and the following fields should be defined. Once the turn-in is **Approved** by the user responsible for the deposit control associated with the tender control, all of the following fields become display-only and the turn-in event cannot be deleted.
- **Tender Type** is the type of tender that has been turned in. This field may only be modified when the **Turn In Status** is **Turned In**.

- **Turn In Amount** is the amount of the **Tender Type** that has been turned in. This field may only be modified when the **Turn In Status** is *Turned In*.
- **Receipt Number** is the ID of the receipt given to the person who turn-in the funds. This field may only be modified when the **Turn In Status** is *Turned In*.
- **Create Date/Time** contains when the turn-in event was created. This is a display-only field.

## Tender Control - Exceptions

This page contains an entry for every exception (i.e., error) associated with the tender control's payments and payment events. Open this page using **Financial, Tender Control** and navigate to the **Exceptions** tab.

Refer to [Exceptions](#) for more information.

### Description of Page

This page contains an entry for every exception (i.e., error) associated with the tender control's payments and payment events.

## Tender Control - Characteristics

To update tender control characteristics, open **Financial, Tender Control** and navigate to the **Characteristics** tab.

**Note:** If your tender control is balanced, you cannot update tender control characteristics.

### Description of Page

The **Characteristics** collection contains information that describes miscellaneous information about the tender control. The following fields display:

<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Sequence</b>	Controls the order in which characteristics of the same type are displayed.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

## Interfacing Payments From External Sources

Most payments are NOT added by an operator using the Payment Event page. Rather, they are interfaced from an external source (e.g., a lock box or a remittance processor).

The base-package provides two interfaces to upload payments, each based on a different method of creating payment events.

Refer to [Distributing A Payment Event](#) for more information about how payment event distribution is handled in the system.

The topics in this section describe how these payment interfaces work.

**Contents**

[Interfacing Payments](#)  
[Interfacing Payments Using Distribution Rules](#)

## Interfacing Payments

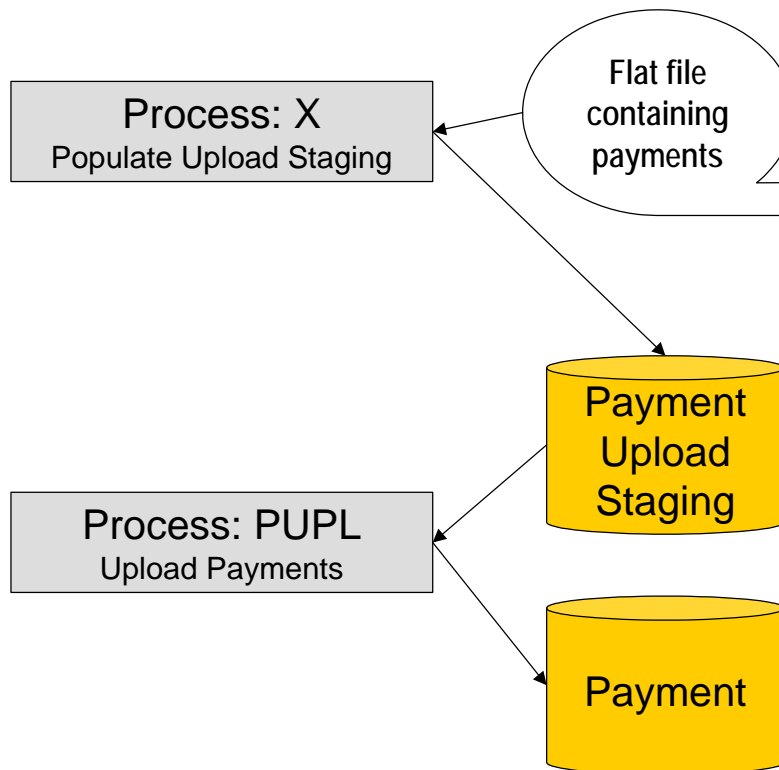
The topics in this section describe how the payment interface using the system default method of creating payment events works.

**Contents**

[Populating The Payment Upload Staging Records](#)  
[PYUP-PRG - Purge Payment Upload Objects](#)  
[Maintaining Deposit Control Staging](#)  
[Payment Upload Staging](#)  
[Payment Upload Exception](#)

### Populating The Payment Upload Staging Records

The following diagram illustrates the processes involved in the uploading of payment into the system.



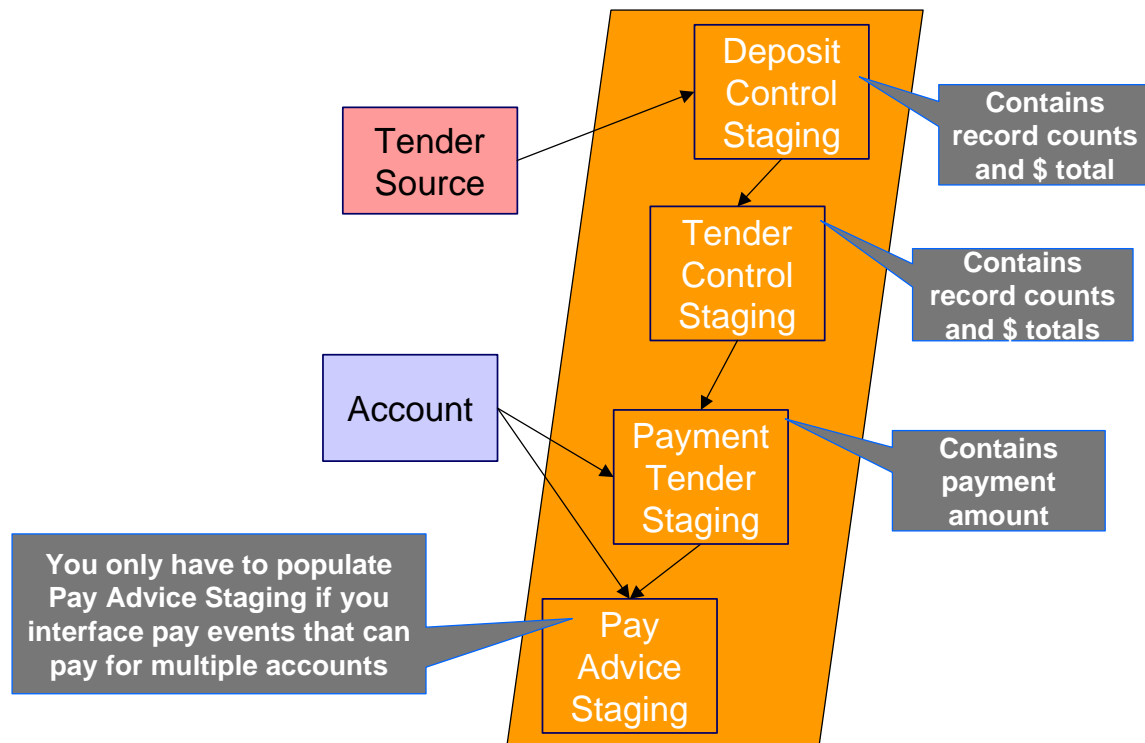
The topics in this section describe each background process referenced above.

**Contents**

[Process X - Populate Payment Upload Staging](#)  
[Process PUPL - Upload Payments](#)

### Process X - Populate Payment Upload Staging

Process X refers to the mechanism used by your organization to populate the various staging tables (shown in the orange section of the following ERD).



The topics in this section describe each of these tables.

#### Contents

[Deposit Control Staging](#)  
[Tender Control Staging](#)  
[Payment Tender Staging](#)  
[Payment Advice Staging](#)

#### Deposit Control Staging

You must create a deposit control staging record for each batch of payments to be uploaded into the system. The name of this table is [CI\\_DEP\\_CTL\\_ST](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
EXT_SOURCE_ID	30	Y	A/N	This must correspond with an external source ID on one of the defined tender sources. Refer to <a href="#">Setting Up Tender Sources</a> for more information.
EXT_TRANSMIT_ID	30	Y	A/N	This is the unique identifier of the transmission from the external source. This must be a unique value for each transmission from the source.
DEP_CTL_STG_ST_FLG	2	Y	A/N	This must be set to 20 (20 is the lookup value that corresponds with the <i>Pending</i> state)

DEP_CTL_ID	10	N	N	Leave this column blank. It will be assigned by the system when it creates a deposit control record.
TRANSMIT_DTTM	15	Y	DateTime	Date and time that the file was transmitted.
CURRENCY_CD	3	Y	A/N	This must be a valid currency code (this would be <i>USD</i> for United States dollars).
TOT_TNDR_CTL_AMT	13.2	Y	N	This column must equal the sum of the payment amounts on the tender control staging records associated with this deposit control staging.
TOT_TNDR_CTL_CNT	10	Y	N	This column must equal the number of tender control staging records associated with this deposit control staging.
LAST_UPDATE_INST	10	N	N	This field is populated during the upload. It is the process scheduler instance ID of the process performing the upload.

You must create one or more [Tender Control Staging](#) for this deposit control staging record.

#### Tender Control Staging

You must create at least one tender control staging record for each batch of payments to be uploaded into the system. The name of this table is [CI\\_TNDR\\_CTL\\_ST](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
EXT_SOURCE_ID	30	Y	A/N	This must correspond with the external source ID on the parent deposit control staging record.
EXT_TRANSMIT_ID	30	Y	A/N	This must correspond with the external transmission ID on the parent deposit control staging record.
EXT_BATCH_ID	30	Y	A/N	This is the unique identifier of the batch of payments in respect of the external transmission ID.
TNDR_CTL_STG_ST_FLG	2	Y	A/N	This must be set to 20 (20 is the translate value that corresponds with the <i>Pending</i> state)
TNDR_CTL_ID	10	N	N	Leave this column blank. It will be assigned by the system when it creates a tender control record.
TOT_TNDR_AMT	13.2	Y	N	This column must equal the sum of the payment amounts on the payment tender staging records associated with this tender control staging.
TOT_TNDR_CNT	10	Y	N	This column must equal the number of payment tender staging records associated with this tender control staging.

You must create one or more [Payment Tender Staging](#) records for this tender control staging record.

### Payment Tender Staging

You must create at least one payment tender staging record for each payment associated with the tender control staging record. The name of this table is [CI\\_PAY\\_TNDR\\_ST](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
EXT_SOURCE_ID	30	Y	A/N	This must correspond with the external source ID on the parent deposit control staging record.
EXT_TRANSMIT_ID	30	Y	A/N	This must correspond with the external transmission ID on the parent tender control staging record.
EXT_BATCH_ID	30	Y	A/N	This must correspond with the external batch ID on the parent tender control staging record.
EXT_REFERENCE_ID	30	Y	A/N	This is the unique identifier of the payment in respect of the external batch ID.
PAY_TND_STG_ST_FLG	2	Y	A/N	This must be set to 10 (10 is the translate value that corresponds with the <i>Pending</i> state)
PAY_TENDER_ID	12	N	N	Leave this column blank. It will be assigned by the system when it creates a tender record.
TENDER_AMT	13.2	Y	N	The amount tendered (i.e., the payment amount).
ACCOUNTING_DT	10	Y	Date	This is the date that should be used for accounting purposes. This should correspond with an open accounting period.
TENDER_TYPE_CD	4	Y	A/N	This must correspond with the prime key of one of your tender types. Refer to <a href="#">Setting Up Tender Types</a> for more information.
CUST_ID	15	Y	A/N	This is the account ID or old account number of the taxpayer tendering the payment. If the system cannot find an account ID or old account number that matches this value, the account ID of the tender source's suspense obligation will be used on the corresponding tender and payment.
MICR_ID	30	N	A/N	This is the MICR ID associated with the payment.
NAME1	40	N	A/N	This is the taxpayer name on the payment.
CHECK_NBR	10	N	A/N	This is the check number on the payment.

### Payment Advice Staging

You need only populate rows on this table if any of the following conditions apply:

- If you need to distribute a payment tender to an account other than that defined with the CUST\_ID on the payment tender staging record, you must create a payment staging record. You may distribute a tender to multiple accounts by creating multiple payment staging records. Note, if you want to distribute the payment tender to the same account, you do NOT need a payment staging record.



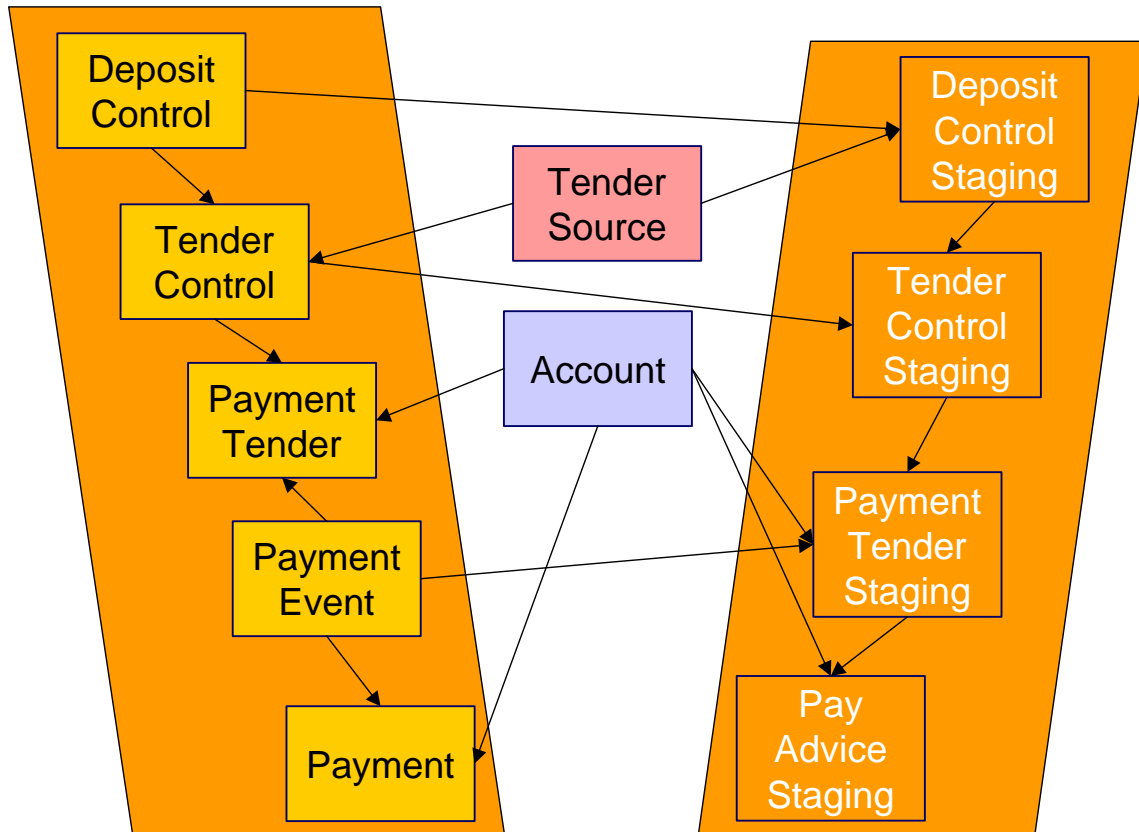
- If you want to restrict a payment to a specific obligation, you must insert a row on this table to indicate the specific the obligation in question. You do this by populating MATCH\_TYPE\_CD with a value that indicates that you are paying for a specific obligation and MATCH\_VALUE with the unique ID of the obligation.
- If you practice [open-item accounting](#), you must insert a row on this table for each to indicate the open-item to which the payment should be matched. Note, because open-item taxpayer typically match payments to bills, you would populate MATCH\_TYPE\_CD with a value to indicate that you are matching by bill ID and MATCH\_VALUE with the unique ID of the bill.

The name of this table is [CI\\_PAY\\_ST](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
EXT_SOURCE_ID	30	Y	A/N	This must correspond with the external source ID on the parent payment tender staging record.
EXT_TRANSMIT_ID	30	Y	A/N	This must correspond with the external transmission ID on the parent payment tender staging record.
EXT_BATCH_ID	30	Y	A/N	This must correspond with the external batch ID on the parent payment tender staging record.
EXT_REFERENCE_ID	30	Y	A/N	This must correspond with the external reference ID on the parent payment tender staging record.
CUST_ID	15	Y	A/N	This is the account ID or old account number of the taxpayer to which the payment should be distributed. If the system cannot find an account ID or old account number that matches this value, the account ID of the payor is used on the corresponding payment. If the payor's account ID is invalid, the tender source's suspense obligation is used.
PAY_AMT	13.2	Y	N	The amount tendered (i.e., the payment amount).
MATCH_TYPE_CD	8	N	A/N	See the description of the MATCH_VALUE field below. Refer to <a href="#">Payments And Match Events</a> for more information about the significance of this field.
MATCH_VALUE	30	N	A/N	MATCH_VALUE and MATCH_TYPE_CD are used in conjunction to indicate that the distribution of the payment should be restricted in some way (i.e., the standard payment distribution algorithm should not be used). MATCH_TYPE_CD indicates how the payment should be distributed (e.g., only distribute to a specific obligation), MATCH_VALUE contains the ID of the restriction (e.g., the obligation ID).  If MATCH_TYPE_CD is specified, it must reference a valid <a href="#">Match Type</a> .

### Process PUPL - Upload Payments

The batch process identified by the batch process ID **PUPL** refers to the background process that loads the contents of the various payment staging records into the various payment event tables. The tables that are populated by this process are shown in the left orange section of the following ERD (the right orange section are the staging tables populated by the process described above)



The topics in this section describe how these tables are populated.

### Contents

Phase 1 - Create Deposit Control

Phase 2 - Create Tender Control

Phase 3 - Create Payment Events, Tenders, Payments and Payment Segments

#### Phase 1 - Create Deposit Control

The following points describe, at a high level, the first phase of the payment upload process:

- PUPL checks that the record counts and money totals of tender control stagings add up to the expected amount on deposit control staging. If not,
  - PUPL sets the status of the deposit control staging to be **Error**. None of the tender controls within the deposit control will be processed until everything adds up. You can fix these on the [Deposit Control Staging](#) page.
  - When PUPL runs next, it will recheck the totals on deposit control stagings that are in **Error** or **Pending**
- If the record and dollar amounts are clean,
  - PUPL creates the corresponding deposit control
  - PUPL sets the status of the deposit control staging to be **In Progress**

#### Phase 2 - Create Tender Control

The following points describe, at a high level, the second phase of the payment upload process:

- PUPL checks that record counts and money totals of payment tender staging(s) adds up to expected amount on tender control staging. If not,
  - PUPL sets the status of the tender control staging to be **Error**. None of the tender controls within the deposit control will be processed until everything adds up for ALL tender controls. You can fix these on the [Tender Control Staging](#) page.

**Note** that the Deposit Control Staging record is **not** updated – its status is unchanged. Neither is the Pay Tender Staging record updated – its status also remains unchanged. Only the Tender Control Staging record is updated to be in **Error**.

- When PUPL runs next, it will recheck the totals of tender control stagings that are in **Error** or **Pending**.
- If the record and dollar amounts are clean,
  - PUPL creates the corresponding tender control.
  - PUPL sets the status of the tender control staging to be **In Progress**.

### Phase 3 - Create Payment Events, Tenders, Payments and Payment Segments

At this point, all deposit control stagings and tender control stagings are in the state of **In Progress**. Next, PUPL starts the upload of payment tender staging and payment advice staging. The following points describe, at a high level, this phase of the payment upload process:

- If the payment tender staging record has a future accounting date, the processing for the record is skipped. This prevents uploaded payments from being created and subsequently frozen until their accounting date is reached. (Some external sources may provide advance notification of payments to be made in the future.) A skipped staging record remains in the **Pending** state until its accounting date is reached.
- PUPL checks money totals of payment advices (if any) adds up to expected amount on payment tender staging.
  - If not, PUPL sets the payment tender staging's status to **Error**.
  - Any errors are written to the [Payment Upload Exception](#) table. You can fix these errors on the [Payment Upload Staging](#) page and change the record's status back to **Pending**.
  - When PUPL runs next, it will recheck the totals of the payment tender staging
- If payment tender staging record is clean:
  - PUPL creates a corresponding payment event, tender, and payment.
  - If the account on payment tender staging is wrong, the account on the corresponding tender will be the tender source's suspense obligation's account. Refer to [Setting Up Tender Sources](#) for more information. Refer to [How To Transfer A Payment From One Account To Another](#) for how to transfer to payment to the correct account.
  - If the account on payment advice is wrong, the account on the corresponding payment will be the account on the payment tender.
  - PUPL distributes the payment(s) amongst the account's obligations, and payment segments are created. Note, the payment could be in error if there are no obligation's for the account (as well as other reasons). Payments in error are written to the [Payment Exceptions](#) table.

- PUPL changes the payment tender staging's status to **Complete**.
- If all payment tender stagings are **Complete**:
  - PUPL changes the tender control staging's status to **Complete**.
  - PUPL changes the deposit control staging's status to **Complete**.
- If there are payment tender staging that are not **Complete**
  - The status of the tender control staging will still be **In progress**.
  - The status of the deposit control staging will still be **In progress**.
- PUPL will attempt to upload the offending payment tender staging records when it next runs.

### **PYUP-PRG - Purge Payment Upload Objects**

**Completed** payment upload staging objects should be periodically purged from the system by executing the [PYUP-PRG](#) background process. This background process allows you to purge all **Completed** payment upload staging objects older than a given number of days.

We want to stress that there is no system constraint as to the number of **Completed** payment upload objects that may exist. You can retain these objects for as long as you desire. However we recommend that you periodically purge **Completed** payment upload objects as they exist only to satisfy auditing and reporting needs.

### **Maintaining Deposit Control Staging**

The Deposit Control Staging page has three purposes:

- You can view historical deposit and tender control staging records associated with uploaded payments.
- You can correct deposit and tender control records that are in error.
- You can add deposit and tender control records to be uploaded by the payment upload background process.

The topics in this section describe this page.

#### **Contents**

[Deposit Control Staging - Main](#)

[Deposit Control Staging - Tender Control Staging](#)

#### **Deposit Control Staging - Main**

This page shows the details of a deposit control staging record.

Refer to [Populating The Payment Upload Staging Records](#) for more information about this record.

Open this page using **Financial, Deposit Control Staging**.

#### **Description of Page**

**External Source ID** corresponds with an external source ID on one of the your tender sources. This should be the unique ID of the source of the interfaced payments. Refer to [Setting Up Tender Sources](#) for more information.

**External Transmit ID** is the unique identifier of the transmission of payments from the external source. This must be a unique value for each transmission from the source.

**Status** shows the state of the deposit control staging records. Potential values are: *Incomplete, Pending, In Progress, Partial Load, Complete, Error*.

**Deposit Control ID** is the system-assigned, unique identifier of the related deposit control. This value is populated after the system creates a deposit control for the upload staging record.

**Transmission Date/Time** are when the information was interfaced into the system.

**Total Tender Controls** must equal the number of tender control staging records associated with this deposit control staging.

**Total Tender Control Amount** must equal the sum of the payment amounts on the tender control staging records associated with this deposit control staging. The **Currency Code** related to the amount is adjacent.

### Deposit Control Staging - Tender Control Staging

This page shows the details of a tender control staging record.

Refer to [Populating The Payment Upload Staging Records](#) for more information about this record.

Open this page using **Financial, Deposit Control Staging, Tender Control Staging**.

### Description of Page

**External Source ID** is the external source ID on the parent deposit control staging record.

**External Transmit ID** is the external transmission ID on the parent deposit control staging record.

The grid that follows contains a row for every tender control staging record linked to the deposit control staging record. The following information is displayed.

<b>External Batch ID</b>	This is the unique identifier of the batch of payments in respect of the external transmission ID.
<b>Status</b>	This is the state of the tender control staging records. Potential values are: <i>Pending, In Progress, Complete, Error</i> .
<b>Tender Control ID</b>	This is the system-assigned, unique identifier of the related tender control. This value is populated after the system creates a tender control for the upload staging record.
<b>Total Tenders Amount</b>	This is the sum of the payment amounts on the payment staging records associated with this tender control staging.
<b>Total Number Of Tenders</b>	This is the number payment tender staging records associated with this tender control staging.

## Payment Upload Staging

The Payment Upload Staging page has three purposes:

- You can view historical payment tender and payment advice staging records associated with uploaded payments.
- You can correct payment tender records that are in error.
- You can add new payment tender and payment advice staging records to be uploaded by the payment upload process.

The topics in this section describe this page.

### Contents

[Payment Upload Staging - Tender Detail](#)

[Payment Upload Staging - Payment Advice](#)

#### Payment Upload Staging - Tender Detail

This page shows the details of a payment tender staging record.

Refer to [Populating The Payment Upload Staging Records](#) for more information about this record.

Open this page using **Financial, Payment Upload Staging, Tender Detail**.

#### Description of Page

**External Source ID** this is the external source ID on the parent tender control staging record.

**External Transmission ID** is the external transmission ID on the parent tender control staging record.

**External Batch ID** is the external batch ID on the parent tender control staging record.

**Ext. Reference ID** is the external source's unique identifier of the payment tender.

**Customer ID** is the account ID or old account number of the taxpayer tendering the payment. If the system cannot find an account ID or old account number that matches this value, the account ID of the tender source's suspense obligation will be used on the corresponding tender and payment.

The **Tender Amount** is the amount tendered (i.e., the payment amount).

**Tender Type** defines the type of tender. Refer to [Setting Up Tender Types](#) for more information.

**MICR ID** is the MICR ID associated with the tender.

**Check Number** is the check number on the payment.

**Name** is the taxpayer's name (as it appeared on the uploaded tender).

**Accounting Date** is the date that should be used for accounting purposes.

**Pay Tender Staging Status** shows the state of the tender control staging records. Potential values are: **Pending**, **Complete**, **Error**.

**Payment Event ID** is the system-assigned, unique identifier of the related payment event. This value is populated after the system creates a payment event for the upload staging record.

## Payment Upload Staging - Payment Advice

If a tender was distributed to taxpayers other than that defined on the Tender Detail page, the taxpayers that the tender was distributed to are defined on this page. This page will not contain information if the tender is distributed to the tender detail's taxpayer.

Refer to [Populating The Payment Upload Staging Records](#) for more information about this record.

Open this page using **Financial, Payment Upload Staging, Payment Advice**.

### Description of Page

**External Source ID** this is the external source ID on the parent tender control staging record.

**External Transmission ID** is the external transmission ID on the parent tender control staging record.

**External Batch ID** is the external batch ID on the parent tender control staging record.

**External Reference ID** is the external reference ID of the payment tender.

The grid that follows is only populated if the tender is distributed to taxpayer(s) other than the tendering taxpayer. The following information is displayed.

<b>Customer ID</b>	This is the account ID or the old account number of the taxpayer to which the payment should be distributed.
<b>Customer Info</b>	If the <b>Customer ID</b> is an account ID that exists in the system, the name of the primary person and the account type of the account are displayed here.
<b>Payment Amount</b>	This is the amount of the tender to be distributed to the taxpayer.
<b>Match Type and Match Value</b>	These fields are only used if the distribution of the payment should be restricted in some way (i.e., the standard payment distribution algorithm should not be used). <b>Match Type</b> indicates how the payment should be distributed (e.g., only distribute to a specific obligation), <b>Match Value</b> indicates the ID of the restriction (e.g., the obligation ID). Valid values of <b>Match Type</b> are <i>obligation</i> .

## Payment Upload Exception

If errors are detected during the payment upload process, a record is written to the payment upload exception table with a message indicating the nature of the severe error.

To view the messages associated with the exception records, schedule the [TD-PYUPL](#) background process. This process generates a To Do entry for every record in the payment upload exception table.

You can fix this error using the [Payment Upload Staging](#) page and change the status of the record from **Error** to **Pending**. When the payment upload process next runs, it attempts to upload this record again.

## Interfacing Payments Using Distribution Rules

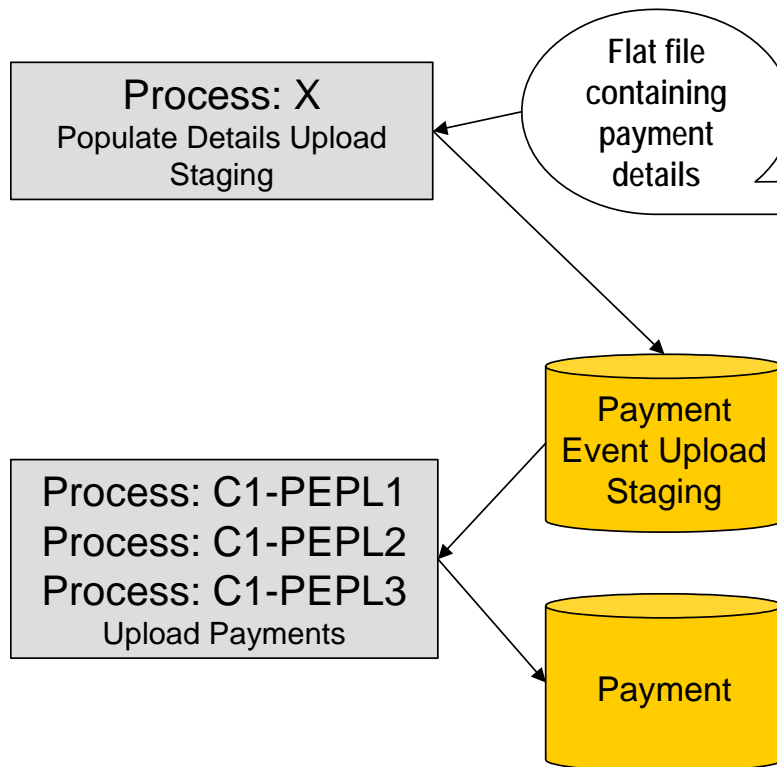
The topics in this section describe how the payment interface using distribution rules works.

### Contents

[Populating The Payment Event Upload Staging Records](#)  
[Payment Event Upload Staging](#)

### Populating The Payment Event Upload Staging Records

The following diagram illustrates the processes involved in the uploading of payment event distribution details into the system.



The topics in this section describe each background process referenced above.

### Contents

[Process X - Populate Payment Event Upload Staging](#)  
[Data Setup Examples of Payment Distribution Details](#)  
[The Lifecycle of a Payment Event Upload Staging Record](#)  
[ToDo Entries Instead of Exception Records](#)  
[Process C1-PEPL1 - Upload Payments \(Step 1\)](#)  
[Process C1-PEPL2 - Upload Payments \(Step 2\)](#)  
[Process C1-PEPL3 - Upload Payments \(Step 3\)](#)

#### Process X - Populate Payment Event Upload Staging

Process X refers to the mechanism used by your organization to populate the payment event upload staging table.



### Payment Event Upload Staging Table

You must create a payment event upload staging record for each payment distribution detail to be uploaded into the system. The name of this table is [CI\\_PEVT\\_DTL\\_ST](#). The following table describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
EXT_SOURCE_ID	30	Y	A/N	This must correspond with an external source ID on one of the defined tender sources. Refer to <a href="#">Setting Up Tender Sources</a> for more information.
EXT_TRANSMIT_ID	30	Y	A/N	This is the unique identifier of the transmission from the external source. This must be a unique value for each transmission from the source.
PEVT_DTL_SEQ	12	Y	N	Unique identifier of the detail record within the transmission. The <a href="#">C1-PEPL1</a> process uses this field to organize the parallel threads.
PEVT_STG_ST_FLG	4	Y	A/N	This must be set to 10 (10 is the lookup value that corresponds with the <i>Incomplete</i> state)
DST_RULE_CD	12	Y	A/N	This must be a valid <a href="#">distribution rule</a> . The distribution rule contains the set of algorithms designed to process the staging detail.
DST_RULE_VALUE	254	Y	A/N	This must be a valid value for the characteristic type defined on the distribution rule.
CURRENCY_CD	3	Y	A/N	This must be a valid currency code (this would be <i>USD</i> for United States dollars).
TENDER_AMT	13.2	Y	N	The amount tendered (i.e., the payment amount).
ACCOUNTING_DT	10	Y	Date	This is the payment date that should be used for accounting purposes. This should correspond with an open accounting period.
TENDER_TYPE_CD	4	Y	A/N	This must be a valid tender type. Refer to <a href="#">Setting Up Tender Types</a> for more information.
CHECK_NBR	10	N	A/N	This is the check number on the payment.
MICR_ID	30	N	A/N	This is the MICR ID associated with the payment tender.
CUST_ID	15	N	A/N	This field may be used to record taxpayer information.
NAME1	30	N	A/N	This field may be used to capture additional payment tender information.
EXT_REFERENCE_ID	30	N	A/N	This field may be used to capture external information associated with the payment tender.
MATCH_TYPE_CD	8	N	A/N	See the description of the MATCH_VALUE field below. Refer to <a href="#">Payments And Match Events</a> for more information about the significance of this field.
MATCH_VALUE	30	N	A/N	MATCH_VALUE and MATCH_TYPE_CD are used in conjunction to indicate that the distribution of the payment should be restricted in some way (i.e., the

				standard payment distribution algorithm should not be used). MATCH_TYPE_CD indicates how the payment should be distributed (e.g., only distribute to a specific obligation), MATCH_VALUE contains the ID of the restriction (e.g., the obligation ID). If MATCH_TYPE_CD is specified, it must reference a valid <a href="#">Match Type</a> .
TNDR_CTL_ID	10	N	A/N	Leave this column blank. It will be assigned by the system when it creates a tender control record.
ACCT_ID	10	N	A/N	This is the tender account. If left blank, the <i>C1-PEPL1</i> process will populate this field by calling the <i>Determine Tender Account</i> algorithm defined on the <a href="#">distribution rule</a> .  Note that this Account ID is not necessarily unique as multiple staging details can reference the same tender account.
PAY_EVENT_ID	12	N	A/N	Leave this column blank. It will be assigned by the system when it creates a payment event record.
PEVT_PROCESS_ID	10	N	A/N	If left blank, the <i>C1-PEPL1</i> process will set this field equal to the Tender Account ID.  This field is used for grouping staging records <b>and</b> for organizing parallel threads (in the <i>C1-PEPL2</i> process). Therefore, it is <b>strongly encouraged</b> that it bears a relationship to the tender account ID.
APAY_SRC_CD	12	N	A/N	This must be a valid auto-pay source code.
EXT_ACCT_ID	50	N	A/N	This is the taxpayer's account number at the financial institution
EXPIRE_DT	10	N	Date	This field is only needed if the Tender Type indicates that an expiration date is necessary (e.g., for a credit card payment)
ENTITY_NAME	64	N	A/N	This is the taxpayer's name in the financial institution's system

### Data Setup Examples of Payment Distribution Details

Typically, each staging record will represent a single payment event – with a corresponding payment tender and payment. However, it is possible to specify complex relationships within a set of staging records. For example, it will be straightforward to define a set of staging records that represent a single tender but multiple payments (to model the single payment tender of a welfare agency which covers payments for multiple accounts). Similarly, it is equally possible to define a set of staging records that represent a single payment but multiple tenders (although not a common requirement).

Each staging record can represent:

- Zero or one new payment events. Typically, each detail staging record will represent a single payment event. However, it is possible to define multiple records for a single payment event. All details for a single payment event are identified by a common value on the staging record: **Pay Event Process ID**.

**Pay Event Process ID.** This field is used for grouping staging records **and** for organizing parallel threads (in the **C1-PEPL2** process). Therefore, it is **strongly encouraged** that it bears a relationship to the tender account ID.

- A partial or one new payment tenders. Typically, each detail staging record will represent a single payment tender. However, it is possible to define multiple staging records for a single pay tender. All details for a single tender are identified by a common payment event ID as well as common tender information (tender type, tender account ID, check number, external reference, etc.).
- A partial or many new payments. Typically, each detail staging record will represent a set of one or many new payments. However, it is possible to define multiple staging records for a single payment. All details for a single payment are identified by a common payment event ID as well as common distribution rule and characteristic value information.

The sections below provide examples of a few of these complex payment event configurations. Note that these examples assume the same distribution rule is referenced in all staging records.

### Contents

[Staging Entry Example 1: One Payment Event, Two Tenders, Two Payments](#)

[Staging Entry Example 2: One Payment Event, One Tender, Two Payments](#)

[Staging Entry Example 3: One Payment Event, Two Tenders, One Payment](#)

#### Staging Entry Example 1: One Payment Event, Two Tenders, Two Payments

Seq.	Pay Event Process ID	Tender Account ID	Tender Ref.	Date	Rule Value	Amount	Tender Type	Check No
1	1234567890	1234567890	112A	1/1/2006	113-54-8978	30.00	Check	101
2	1234567890	1234567890	112B	1/1/2006	575-40-3030	40.00	Check	102

#### Staging Entry Example 2: One Payment Event, One Tender, Two Payments

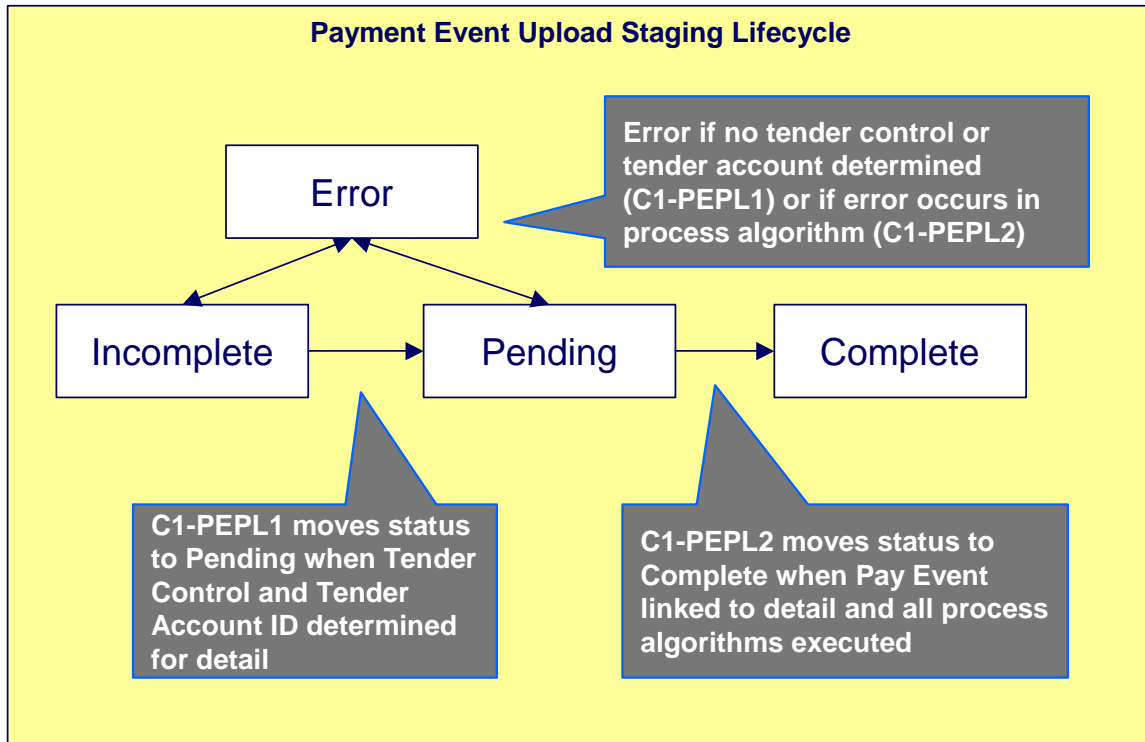
Seq.	Pay Event Process ID	Tender Account ID	Tender Ref.	Date	Rule Value	Amount	Tender Type	Check No
1	1234567890	1234567890	112A	1/1/2006	113-54-8978	30.00	Check	101
2	1234567890	1234567890	112A	1/1/2006	575-40-3030	40.00	Check	101

#### Staging Entry Example 3: One Payment Event, Two Tenders, One Payment

Seq.	Pay Event Process ID	Tender Account ID	Tender Ref.	Date	Rule Value	Amount	Tender Type	Check No
1	1234567890	1234567890	112A	1/1/2006	575-40-3030	30.00	Check	101
2	1234567890	7878787870	888Q	1/1/2006	575-40-3030	40.00	Check	9872

### The Lifecycle of a Payment Event Upload Staging Record

The following diagram shows the possible lifecycle of a payment event upload staging record.



- **Incomplete.** A payment event staging record is initially created in **incomplete** state. The **C1-PEPL1** process sets it to **pending** once it links it to a tender control and determines its tender account.
- **Pending.** The **C1-PEPL2** process sets a **pending** record to **complete** once all processing logic is executed and a payment event is linked to it.
- **Complete.** When processing of the staging record is complete the record is in the **complete** state. This is a final state.
- **Error.** A payment event staging record may be set to **Error** from **Incomplete** or **Pending** states by the **C1-PEPL1** and **C1-PEPL2** processes respectively.

Refer to [ToDo Entries Instead of Exceptions](#) for more information on how ToDo entries are used to capture processing errors.

### ToDo Entries Instead of Exception Records

Instead of creating an exception record for staging records in the **Error** state the **C1-PEPL1** and **C1-PEPL2** background processes create To Do entries and link them to the offending staging records.

Each process determines the [To Do type](#) to use for reporting errors by looking for the To Do Type defined with this process as its creation process.

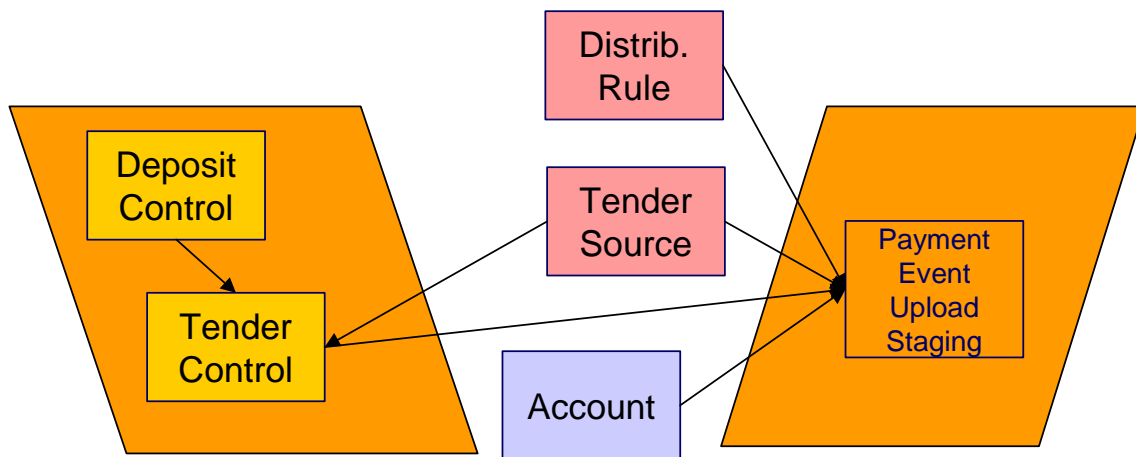
### Process C1-PEPL1 - Upload Payments (Step 1)

The batch process **C1-PEPL1** refers to the first of three background processes that load the contents of the payment event upload staging records into the various payment tables.

The responsibility of this process is to transition the status of the staging records from **Incomplete** to **Pending**. The status of a staging record must remain **Incomplete** until:

- It is linked to a tender control. This process creates new deposit and tender control records, and then updates the payment event upload staging records with the corresponding **Tender Control ID**.
- The **Tender Account ID** field is populated. The **Determine Tender Account** algorithm defined on the [distribution rule](#) is called and the returned account ID is posted on the staging record.
- The **Pay Event Process ID** field is populated. If left blank this field is set equal to the **Tender Account ID**.

The following diagram and the sections below describe at a high level the processing phases of the **C1-PEPL1** background process.



### Contents

Phase 1 - Create Tender Control

Phase 2 - Determine Tender Account

### Phase 1 - Create Tender Control

**Note.** This step cannot be bounded by thread range but must execute across the entire population of staging details. Therefore this step in the process is designed so that all parallel threads attempt to execute it at the same time but only one thread succeeds to avoid creating duplicate tender controls.

This step creates all tender controls required by the upload records as follows:

- For each distinct tender source transmissions represented within the **incomplete** set of upload staging details a deposit control, a deposit tender and a tender control are created in an **open** state. Note that a deposit tender record is only created if the tender amount is not zero.
- If an error occurs at this stage
  - A designated staging record for the transmission group (one is picked at random) is set to **Error** and a ToDo entry is created and linked to it to capture the error message that applies for the whole transmission group. Other records in the group remain **incomplete**.

- The process stops.

**Group Error.** This technique allows for an easier recovery from a setup error that may affect a large volume of records in a single transmission. Capturing the error only on a single designated record requires only this record to be set back to **Incomplete** once the setup issue is corrected. It is important to note that the transmission group is not processed if at least one of the records in the group is in **Error** status.

- Once a transmission group of records is fully processed, a ToDo cleanup processing takes place to complete ToDo entries previously raised for its designated staging record.

Refer to [ToDo Entries Instead of Exceptions](#) for more information on how ToDo entries are used to capture processing errors.

### Phase 2 - Determine Tender Account

After all tender controls have been created the second step attempts to transition all **incomplete** records to the **pending** state.

Each **incomplete** staging record is processed as follows:

- If not yet associated with a tender control ID, the process looks for an **Open** tender control that matches the batch code, batch number and external source ID and links it to the staging record. An error is raised if a matching tender control is not found.
- Execute the **Determine Tender Account** algorithm defined on the distribution rule and populate the record with the returned account ID. Note that the algorithm is called even when the tender account is populated to provide for a potential override of the initial value when necessary. An error is raised if a tender account may not be determined.
- If not already populated, the **Pay Event Process ID** is set equal to the Tender Account ID. The **C1-PEPL2** process uses this field to organize the parallel threads and to group multiple staging details into a single payment event.
- The staging record is moved to **pending** state.
- If any errors occur set the record to **Error** and create a To Do entry for the error message.
- If no error occurred, a ToDo cleanup processing takes place to complete ToDo entries previously raised for the staging record.

Refer to [ToDo Entries Instead of Exceptions](#) for more information on how ToDo entries are used to capture processing errors.

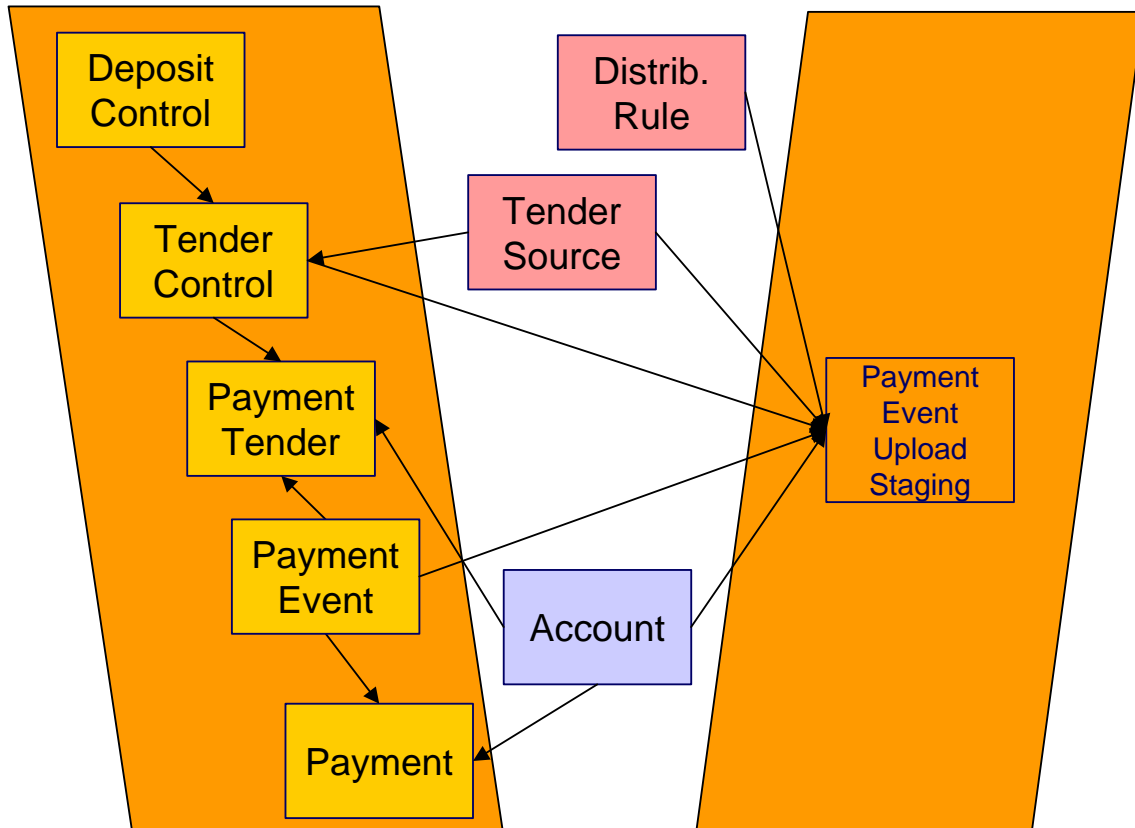
**Note.** This step is designed to support execution in parallel threads based upon the sequence number portion of the staging table prime key.

### Process C1-PEPL2 - Upload Payments (Step 2)

The batch process **C1-PEPL2** refers to the second of three background processes that load the contents of the payment event upload staging records into the various payment tables.

The responsibility of the **C1-PEPL2** process is to create payment events, payment tenders and payments and transition the corresponding staging records from **Pending** to **Complete**.

The following diagram and the section below describe at a high level the processing phases of the **C1-PEPL2** background process.



Each distinct group of **pending** staging records associated with the same external source ID, external transmit ID, accounting date, and pay event process ID is processed as follows:

- A payment event is created for the group and stamped on each of its records.
- A payment tender is created for each distinct set of staging records having the same tender account, tender type and other tender information fields, except for the tender amount.
- The **Create Payment** algorithm is then called for each distinct group of staging records having the same tender account, [distribution rule](#) and rule value providing it with the total amount for the group.
- The staging record is moved to **pending** state.
- If any error occurs a designated staging record for the payment event group (one is picked at random) is set to **Error** and a To Do entry is created and linked to it to capture the error message that applies for the whole group. Other records in the group remain **incomplete**.

**Group Error.** This technique allows for an easier recovery from an error that affects all staging records for a single payment event. Capturing the error only on a single designated record requires only this record to be set back to **pending** once the issue is corrected. It is important to note that the whole set of records is not processes if at least one of the records in the group is in **Error** status.

- If no error occurred, a To Do cleanup processing takes place to complete To Do entries previously raised for the designated staging record.

Refer to [ToDo Entries Instead of Exceptions](#) for more information on how To Do entries are used to capture processing errors.

**Note.** This process is designed to support execution in parallel threads based upon the payment event process ID field.

### Process C1-PEPL3 - Upload Payments (Step 3)

The batch process **C1-PEPL3** refers to the last of three background processes that load the contents of the payment event upload staging records into the various payment tables.

The responsibility of the **C1-PEPL3** process is to update the status of the related deposit and tender controls from **open** to **balanced**.

Each distinct tender control for which all associated staging records are in **complete** status is processed as follows:

- The tender control is set to **balanced**.
- The deposit control is set to **balanced**.

**Note.** This process is designed to support execution in parallel threads based upon the Tender Control ID field.

## Payment Event Upload Staging

The Payment Event Upload Staging page has three purposes:

- You can view historical payment event upload staging records associated with uploaded payments.
- You can correct payment event upload staging records that are in **error**.
- You can add new payment event upload staging records to be uploaded by the payment event upload processes.

The topics in this section describe this page.

### Payment Event Upload Staging - Main

This page shows the details of a payment event upload staging record.



Refer to [Populating The Payment Event Upload Staging Records](#) for more information about this record.

Open this page using **Financial, Payment Event Upload Staging, Main**.

### Description of Page

**External Source ID** corresponds with an external source ID on one of the your tender sources. This should be the unique ID of the source of the interfaced payments. Refer to [Setting Up Tender Sources](#) for more information.

**External Transmit ID** is the unique identifier of the transmission of payments from the external source. This must be a unique value for each transmission from the source.

**Sequence** is the identifier of the record within the transmission. The **C1-PEPL1** process uses this field to organize the parallel threads.

**Accounting Date** is the payment date that should be used for accounting purposes.

**Distribution Rule** is the rule by which the payment detail is to be processed. A [default distribution rule](#) is displayed if you have set one.

**Rule Value** is a value associated with the payment and expected by the distribution rule.

**Match Type** and **Match Value** are only used if the distribution of the payment should be restricted in some way (i.e., the standard payment distribution algorithm should not be used). **Match Type** indicates how the payment should be distributed (e.g., only distribute to a specific obligation), **Match Value** indicates the ID of the restriction (e.g., the obligation ID).

**Payor Account ID** is the tender account. If not populated the **C1-PEPL1** process populates this field by calling the **Determine Tender Account** algorithm defined on the [distribution rule](#).

**Tender Amount** is the amount tendered (i.e., **the** payment amount).

**Currency Code** is the currency of the tendered amount. This should be the same as currency defined on the tender source.

**Tender Type** defines the form or remittance (e.g., cash, check, etc.). Note that the **Tender Type** defaults from the [installation record](#).

**Check Number** is the check number on the payment.

**MICR ID** is the value of the magnetic ink character recognition (MICR) line on the payment.

**External Reference ID** may be used to record external information associated with the payment tender.

**Customer ID** may be used to record additional taxpayer information.

**Name** may be used to record additional payment tender information.

**Tender Control ID** is the tender control associated with the payment. This field is should typically be left for the **C1-PEPL1** process to populate.

If the **Tender Type** is associated with an [automatic payment](#), the following section appears:

The system attempts to default automatic payment information from the [account's auto-pay](#) option if the tender type is the same as the tender type on the account's auto-pay source and if the auto pay option is effective on the payment date. If the system is unable to default information, you must specify the source of the funds and the taxpayer's account number / credit card number at the financial institution.

- **Auto Pay Source Code** is the financial institution / credit card company that receives the automatic payment request.
- **External Account ID** is the taxpayer's account number at the financial institution.
- **Expires On** is only needed if the **Tender Type** indicates that an expiration date is necessary (e.g., for a credit card payment).
- **Name** is the taxpayer's name in the financial institution's system.

**Pay Event Process ID** is used to group multiple staging records into a single payment event. If not populated, the **C1-PEPL1** process sets this field equal to the **Payor Account ID**.

**Pay Event Staging Status** shows the state of the staging record. Refer to [The Lifecycle of a Payment Event Upload Staging](#) for a state transition overview.

**Payment Event ID** is the system-assigned, unique identifier of the related payment event. The **C1-PEPL2** process populates this field when it creates a payment event for the upload staging record. You can use this field to navigate to the payment event page.

If a staging record is in **Error** state then the error message associated with the [corresponding To Do entry](#) is displayed.

# Adjustments

In this section, we describe when and how to adjust an obligation's balance.

## Contents

- [The Big Picture Of Adjustments](#)
- [Maintaining Adjustments](#)
- [How and When To Use An Adjustment](#)
- [Interfacing Adjustments From External Sources](#)

## The Big Picture Of Adjustments

The topics in this section provide background information about a variety of adjustment issues.

We strongly recommend familiarizing yourself with the topics described in [The Financial Big Picture](#) to fully appreciate the place of an adjustment in the system's financial architecture. In particular, refer to [Setting Up Adjustment Types](#) and [Setting Up Adjustment Type Profiles](#).

## Contents

- [Adjustments - Current Balance versus Payoff Balance](#)
- [Canceling Adjustments](#)
- [Transfer Adjustments](#)
- [Calculated Adjustments](#)
- [Adjustments and Penalty and Interest](#)
- [Adjustment Amount May Be Positive, Negative Or Zero](#)
- [Adjustment Type Controls Everything](#)
- [An Adjustment May Affect More Than Just Taxpayer Balances](#)

## Adjustments - Current Balance versus Payoff Balance

Adjusting how much a taxpayer owes involves changing an obligation's payoff balance and/or current balance by creating an adjustment. In this section we describe these two balances.

**Warning!** If you do not understand the difference between payoff balance and current balance, refer to [Current Amount versus Payoff Amount](#).

## Contents

- [When Current Balance Equals Payoff Balance](#)
- [When Current Balance Differs From Payoff Balance](#)
- [Adjustment Type And Balances](#)

## When Current Balance Equals Payoff Balance

For most obligations, payoff balance and current balance are always the same (or in colloquial speech - the amount the taxpayer thinks they owe equals what they really owe). In this situation, an adjustment is easy: both payoff balance and current balance are adjusted by the same value.

Let's run through a typical example. The values in the payoff balance and current balance columns reflect the amount due after the financial transaction has been applied (i.e., the running balance):

Date	Financial Transaction	Payoff Balance	Current Balance
1-Jan-99	Bill: \$125	125	125
15-Jan-99	Payment: \$150	-25	-25
2-Feb-99	Bill: \$175	150	150
14-Feb-99	Payment: \$150	0	0
3-Mar-99	Bill: \$200	200	200
15-Mar-99	Payment: \$150	50	50
2-Apr-99	Bill: \$225	275	275

As you can see, payoff balance and current balance are always in sync.

### When Current Balance Differs From Payoff Balance

For some obligations, payoff balance and current balance differ (or in colloquial speech - the amount the taxpayer thinks they owe differs from what they would owe if they wanted to payoff their account).

The next table shows several examples for tax processing where the current balance differs from the payoff balance.

Obligation Type	Payoff Balance	Current Balance
Taxpayer submits a return submitted prior to the due date	The payoff balance includes the full debt minus any payments.	Until the debt is due, the current amount reflects that no debt is due (credit balance results from any payment).
Real property	Includes the full outstanding debt amount for the obligation period.	Includes only the debt due at the current time.
Taxpayers with a payment agreement	Includes the full outstanding debt amount.	References the current payment portion due.

### Adjustment Type And Balances

When you create an adjustment, you must define its adjustment type. The adjustment type controls how payoff balance and current balance are affected by the adjustment amount.

You can only pick adjustment types that make sense for the obligation.

The various types of adjustments that may be linked to an obligation are controlled by the adjustment profiles defined on the obligation's obligation type.

Refer to [Adjustment Type Controls Everything](#) for more information.

## Canceling Adjustments

You may cancel any frozen adjustment. Canceling the adjustment creates another financial transaction that reverses the financial effects of the original adjustment. The impact of the cancellation appears on the taxpayer's next bill. You may view both the original financial transaction and its cancellation on the Adjustment page.

## Transfer Adjustments

A convenient mechanism exists to transfer moneys between two obligations. The net effect of such a request is the creation of two adjustments, each of which is linked to the other. Both adjustments are created together, frozen together, and posted to the GL together. This is useful because you can't create one side of the transfer and without the other. Refer to [How To Create A Transfer Adjustment](#) for more information.

**Inter or Intra Account.** It's important to be aware that the transfer can be inter or intra account (i.e., the accounts on the two adjustments may be different).

## Calculated Adjustments

An adjustment can calculate the adjustment amount using an algorithm. A calculated adjustment can be used to calculate:

- Sales or other taxes for a base amount
- Any charge amount based on a user-supplied quantity, such as feet, miles, hours, etc.
- Others.... Any calculation that can be made by a rate can be applied to an adjustment.

When you create a calculated adjustment, you may be asked to supply a base amount. The user may supply the base amount or the amount could be [defaulted](#) from the adjustment type and possibly overridden by the user prior to calculating the adjustment amount.

When the adjustment is generated, the calculation algorithm may use the base amount and calculation date. The base package algorithm calls the rate application and returns the calculated amount with calculation lines and GL distribution codes.

An adjustment's adjustment type controls whether the adjustment is calculated and the algorithm that performs the calculation.

For more information, refer to [Adjustment Type Controls Everything](#) and [Setting Up Calculated Adjustment Types](#).

## Adjustments and Penalty and Interest

If your organization calculates penalty and interest (P&I) on outstanding debt, creating adjustments will typically impact the penalty and interest calculations. Most such adjustments are created as part of another business process, such as posting a tax form, where bringing P&I up to date is part of the business process.

For various reasons, the base product does not provide functionality to bring P&I up to date when an adjustment is created manually via the adjustment page. If your organization requires the ability to create a manual adjustment and bring P&I up to date, a special business process should be designed to orchestrate this.

Refer to [Adjustments and Updating P&I](#) for more information.

## Adjustment Amount May Be Positive, Negative Or Zero

An adjustment's amount may be positive, negative or zero:

- A positive amount causes the taxpayer's balance(s) to increase.
- A negative amount causes the taxpayer's balance(s) to decrease.
- A zero amount will not affect the taxpayer's balance(s). A zero amount is odd, but necessary when you need to use an adjustment to correct the GL distribution. Refer to [How To Use An Adjustment To Change Amounts Booked In Your GL](#) for more information.

## Adjustment Type Controls Everything

When you create an adjustment, you must define its adjustment type. The topics in this section describe how adjustment type controls the behavior of an adjustment.

### Contents

- [Controls Which Balance\(s\) Are Affected](#)
- [Defines The GL Account Affected By The Adjustment](#)
- [Defaults The Adjustment Amount](#)
- [Calculates The Adjustment Amount](#)
- [Controls The Interface To A/P and Income Statement Reporting](#)
- [Controls If The Adjustment Requires Approval](#)
- [Controls Information Printed On The Bill](#)

### Controls Which Balance(s) Are Affected

The adjustment type's financial transaction (FT) algorithm controls how payoff balance and current balance are affected by the adjustment amount.

### Defines The GL Account Affected By The Adjustment

Most adjustments affect the general ledger (GL) in some way. The following points describe the source of these GL accounts.

- For normal adjustments (i.e., non-calculated adjustments that affect a single obligation), there is a single accounting entry generated:
  - One side of the accounting entry is taken from the distribution code on the obligation type of the obligation affected by the adjustment. For example, if you are adjusting the payoff balance on a normal obligation, the A/R account is constructed from the distribution code on the obligation's obligation type.
  - The other side of the accounting entry is taken from the distribution code on the adjustment's adjustment type.

- For transfer adjustments (i.e., adjustments used to transfer moneys between two obligations), there are two accounting entries generated - one for the “from” side and one for the “to” side. Each adjustment carries its own set of balanced GL accounting details.
  - For each adjustment, one side of the entry is taken from the distribution code on the obligation type of the obligation affected by the adjustment (just like for normal adjustments).
  - The other sides of both accounting entries have the same GL account. This account should be the intermediate clearing GL account that is to be used for the transfer. The source of this clearing GL account is the distribution code on the adjustment type used to transfer the funds.
- For calculated adjustments (i.e., adjustments where rates may be applied to calculate the adjustment amount), there is a single accounting entry:
  - For each adjustment, one side of the entry is taken from the distribution code on the obligation type of the obligation affected by the adjustment (just like for normal adjustments).
  - The other side of the entry is taken from the distribution codes that resulted from the rate calculation if the adjustment financial transaction algorithm is set up to use the calculation lines as the distribution code source. If the adjustment financial transaction algorithm is set up to use the adjustment type as the distribution code source, the other side of the accounting entry is taken from the distribution code on the adjustment type.

**Not all adjustments affect the GL.** As a general rule of thumb, only those adjustments that affect an obligation's payoff balance affect the GL.

## Defaults The Adjustment Amount

The adjustment type may default the adjustment amount in one of the following ways:

- The adjustment type may specify a default amount. This would be used for those adjustment types that have a standard charge for all taxpayers that receive this adjustment, for example a non-sufficient funds charge.
- The adjustment type may specify a default adjustment amount algorithm. This would be used for those adjustment types that have a charge that varies based on other factors. For example, a non-sufficient funds charge may be based on a taxpayer's credit rating.

When an amount is defaulted onto a new adjustment it may be overridden by a user.

## Calculates The Adjustment Amount

Some adjustment types [calculate the adjustment](#) amount. If the adjustment type uses a calculated amount, a base amount is passed to the rate application, which then returns the calculated amount with calculation lines and GL distribution codes.

The adjustment type's generate adjustment algorithm controls which rate is applied to the base amount. A user supplied calculation date controls which version of the rate is used. The user may supply the base amount or it may be [defaulted](#) from the adjustment type and possibly overridden by the user prior to calculating the adjustment amount.

## Controls The Interface To A/P and Income Statement Reporting

If the adjustment type is associated with a payment (e.g. a refund) of money to a taxpayer, the adjustment type indicates such with a reference to an A/P request type.

When an adjustment that references an A/P request type is **frozen**, an A/P download request record is created. This record is the interface request to ask your A/P system to cut a check. This interface record is marked with a batch process ID and run number.

- The batch process ID is the process responsible for creating the flat file that contains check request that is interfaced to your account's payable system. The batch process ID is defined on [Installation Options – Financial](#). The base package is supplied with a skeletal background process (referred to by the process ID of **APDL**) that must be populated with logic to format the records in the format compatible with your accounts payable system.
- The run number is the batch process ID's current run number.

Refer to [Accounts Payable Check Request](#) for more information.

If the resultant check needs to be reported for income tax purposes under a specific income statement category, the category is also specified on the adjustment type. The income statement category is in turn interfaced to the A/P system (the system does NOT manage income statement reporting).

## Controls If The Adjustment Requires Approval

If an adjustment's adjustment type references an approval profile, the system will not allow the initiating user to freeze the adjustment. Rather, the initiating user can submit the adjustment for approval. When an adjustment is submitted for approval, the system determines the necessary approval levels and notifies the first approver. The system will freeze the adjustment when last approver approves the adjustment (if the adjustment's adjustment type allows it to be frozen prior to the completion of the next bill).

**Only online adjustments are subject to approval.** The system assumes that no approval is necessary for adjustments created by batch processes even those whose adjustment type references an approval profile.

Refer to [The Big Picture of Adjustment Approvals](#) for more information.

## Controls Information Printed On The Bill

If the adjustment appears on the taxpayer's next bill, the verbiage is specified on the adjustment type (and may NOT be overridden on the adjustment).

## An Adjustment May Affect More Than Just Taxpayer Balances

When an adjustment is frozen or cancelled, a taxpayer's current and payoff balances are affected. However, several other objects may be affected when such events occur. Refer to [Obscure Things That Can Happen](#) for more information.



## Maintaining Adjustments

---

An adjustment is used to change the amount of debt stored on an obligation. The topics in this section describe the pages on which adjustments are maintained.

For more information about adjustments, refer to [The Big Picture Of Adjustments](#).

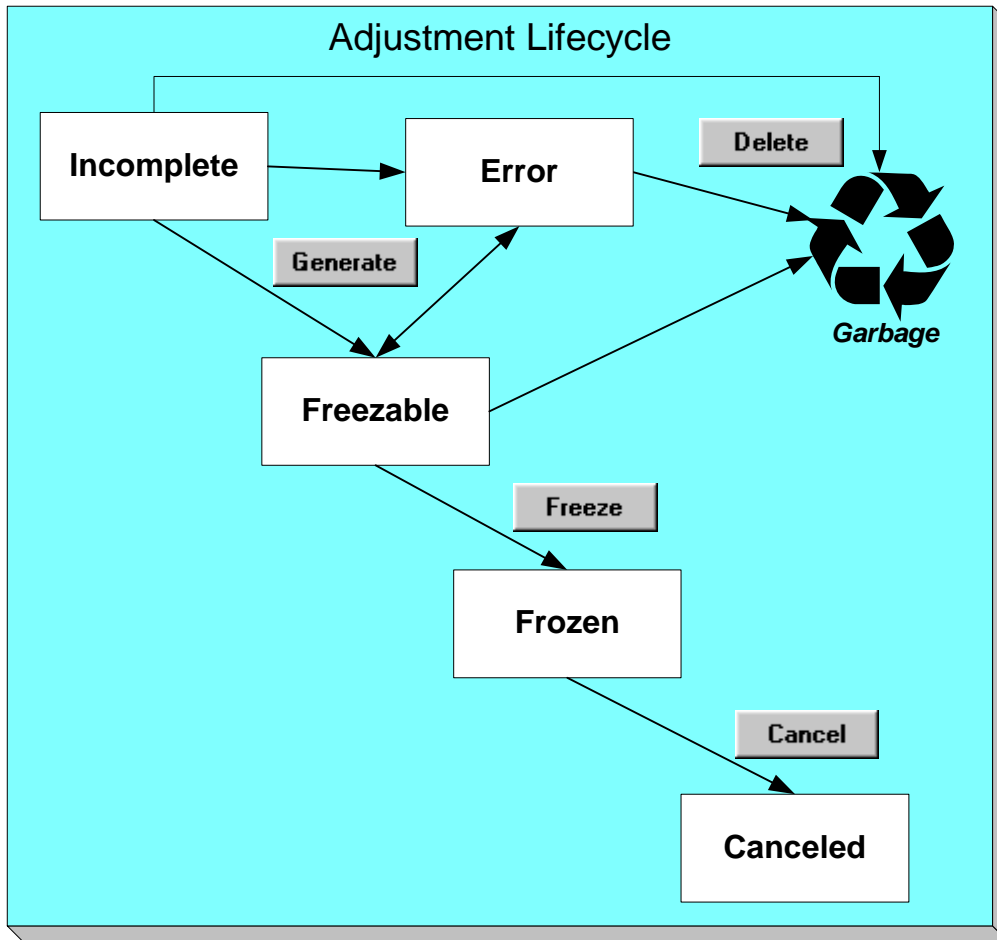
### Contents

- [The Lifecycle Of An Adjustment](#)
- [Adjustments - Main Information](#)
- [Adjustments - Characteristics](#)
- [Adjustments - Transfer Adjustment](#)
- [Adjustments - A/P Request](#)
- [Adjustments - Approval](#)
- [Financial - Adjustment Calculation Line Characteristics](#)

## The Lifecycle Of An Adjustment

The following diagram shows the possible lifecycle of an adjustment.

**Warning!** This diagram only makes sense in the context of the page used to maintain Adjustments. Refer to [Adjustments - Main Information](#) for the details.



Adjustments are initially created in the **Incomplete** state. Adjustments in this state don't have a financial transaction. This means, you can change the adjustment type and amount at will.

Click **Generate** to generate a financial transaction for the adjustment. The financial transaction contains the adjustment's effect on the general ledger and on the taxpayer's payoff and current balances. If the adjustment is calculated, the algorithm on the adjustment type's Generate Adjustment event controls how the adjustment is calculated. If the adjustment is calculated, you must specify the calculation date, which is passed as a parameter to the calculation algorithm and is used for calculations that are effective dated (e.g., rate version or rate factor value). The financial transaction (FT) algorithm defined on the adjustment type Adj. Financial Transaction event controls how the financial transaction is constructed. For calculated adjustments, the distribution code may be taken from the adjustment type or the calculation lines.

After generating the financial transaction, the adjustment becomes **Freezable**. While in this state, you may change the obligation, adjustment type and amount at will. However, if you change this information, you will have to regenerate the financial transaction (by clicking the Generate button).

In the very rare situation when the system cannot generate the financial transaction because of inconsistent setup data, the adjustment is moved to the **Error** state. You may regenerate the financial transaction after correcting the source of the error. You may also delete such an adjustment.

Click the delete button to physically remove an **Incomplete**, **Error** or **Freezable** adjustment from the database.

Click **Freeze** to freeze the adjustment and its financial transaction. After doing this, the adjustment's state becomes **Frozen**. While in this state, you cannot change the adjustment's type or amount, but you may change:

- When the adjustment starts aging.
- Whether the adjustment appears on a taxpayer's bill.
- The accounting date used to derive the general ledger accounting period(s) to which the financial transaction is booked.

**Adjustments may require approval.** If the adjustment type referenced on the adjustment has an approval profile, the Freeze button is replaced with a Submit for Approval button. If an adjustment is currently being approved, all action buttons are disabled because the adjustment must be either approved or rejected using the [Adjustments - Approval](#) page. Refer to [The Big Picture of Adjustment Approvals](#) for more information.

If you need to remove the financial effects of an adjustment, click **Cancel**. Canceling an adjustment causes the generation of another financial transaction. This new financial transaction reverses the financial impact of the original adjustment.

## Adjustments - Main Information

The Description of Page section that appears below simply describes the fields on this page. Refer to [How and When To Use An Adjustment](#) for instructions describing how to perform common maintenance functions.

The Main page contains core adjustment information. Open this page using **Financial, Adjustment**.

**Warning!** All adjustments reference a very important field that controls much validation and processing; this field is called Adjustment Type. Take special care when adding a new adjustment to specify the appropriate Adjustment Type as it affects how the adjustment appears on the taxpayer's bill, how the adjustment is reflected in your general ledger, and much more. After the adjustment is frozen, you may not change its Adjustment Type.

### Description of Page

**Adjustment Info** contains a concatenation of the adjustment amount, adjustment type and status.

**Formatting may be performed by a plug-in.** The contents of **Adjustment Info** may be formatted by a plug-in algorithm on the [Adjustment Type](#). Refer to the base package's [C1-ADT-INFO](#) for an example. If such an algorithm is not plugged-in on the Adjustment Type, the system looks for a corresponding algorithm on the [installation record](#). Refer to the base package's [C1-ADI-INFO](#) for examples. If you prefer different formatting logic, your system administrator should configure the system appropriately.

**Adjustment ID** is the system-assigned, unique identifier of the adjustment.

**Account ID** is the account to which the adjustment is linked. The name of the main person on the account appears next to the account ID.

Use **Obligation ID** to define the obligation whose value needs to be adjusted. Basic information about the obligation appears adjacent.

**Location** is a display-only field that shows the location associated with the obligation, if any.

Indicate the **Adjustment Type**. This field is very important as it controls numerous aspects of the adjustment's impact on the taxpayer's balance and your general ledger. This field is gray after the adjustment is frozen.

**You can only choose certain adjustment types.** The obligation's obligation type has a collection of valid adjustment profiles. You may only reference adjustment types that are listed in one of the adjustment type profiles linked to the obligation type.

Enter the **Amount** of the adjustment. If the adjustment type is a calculated adjustment, the **Calculated Amount** and **Calculation Date** are displayed. The values of the calculated amount and calculation date are displayed after the adjustment is generated. The calculated amount shows the result of the generate adjustment algorithm. The calculation date is specified when you click Generate for a calculated adjustment type. It is used by the generate adjustment algorithm for any calculations that are effective dated (e.g., rate version or rate factor value).

**Default note.** The adjustment amount defaults based on the adjustment type. When you change the adjustment type, the amount changes accordingly. You may change the adjustment amount after it is defaulted.

You may not change an adjustment's **Adjustment Status** directly. Rather, you use the buttons in the **Adjustment Actions** area. Refer to [The Lifecycle Of An Adjustment](#) for the details.

If the status is **Canceled**, the **Cancel Reason** is displayed.

Use the **Comments** to describe anything unusual about the adjustment.

The **Creation Date** defines the date on which the adjustment was created.

If the adjustment is subject to approval, a message indicating such appears. Clicking this message navigates the user to the **Approvals** tab. Refer to [The Big Picture of Adjustment Approvals](#) for more information.

The financial transaction (FT) grid contains the financial transactions associated with the adjustment. It only contains information after the adjustment is frozen. If the adjustment is canceled, a second row appears showing the details of the cancellation FT.

- **Financial Transaction ID** is the system-assigned unique identifier of the FT. Click the go to button to transfer to the financial transaction. On this page you can change certain aspects of the FT in question.
- **Effective Date** is the date the FT starts aging.
- **Accounting Date** is the date the system uses to determine the FT's accounting period in your general ledger.
- **Current Amount** contains the FT's effect on the obligation's current balance.
- **Payoff Amount** contains the FT's effect on the obligation's payoff balance.

- **Bill ID** is the bill on which the FT appears (if it has been swept onto a bill). Click the adjacent go to button to transfer to the bill on which the FT appears. Note: an FT is linked to a bill the next time a bill is completed for the obligation's account.
- **Group FT ID** is the identifier of the main tax liability assessment's FT. Refer to [Group Financial Transactions](#) for more information.

The **Calculation Lines** grid contains the details of the calculations associated with a [calculated adjustment](#). It only appears if the rate that calculated the adjustment amount created at least one calculation line. One row exists for every calculation involved in the process. This information is for audit purposes only and cannot be modified.

The following information is displayed in the grid:

- If at least one of the calculation lines has characteristics, **Calc Line Char** displays go to buttons, allowing you to go to characteristics that are linked to a specific adjustment calculation line. This column is only displayed if at least one of the calculation lines has characteristics.
- **Sequence** is the system-assigned unique identifier of the calculation detail row.
- **Description on Bill** is the information about the calculation line that appears on the taxpayer's bill.
- **Calculated Amount** is the calculated amount associated with the calculation line.
- The **Print** switch controls whether information about this line will print on the taxpayer's bill.
- The **Appears in Summary** switch defines if this line's amount also appears on a summary line. This switch plays a part at bill print time – those lines that appear in a summary print in the left dollar column, those that don't appear in a summary print in the right dollar column.  
Bill segments created by applying a rate have this switch turned on if the corresponding rate component is summarized on a summary rate component.
- **Unit of Measure (UOM)** is the unit of measure of the rate quantity priced on the calculation line.
- **Time of Use (TOU)** is the time-of-use code of the rate quantity priced on the calculation line.
- **RQI** is the rate quantity identifier of the rate quantity priced on the calculation line.
- **Billable Rate Quantity** is the rate quantity priced on the calculation line. This quantity differs from the measured consumption if there are RQ rules or register rules in effect.
- **Base Amount** is used by calculation lines (e.g. taxes) that are cross-referenced to other calculation lines and whose value(s), therefore, depend on the amounts calculated by those other lines. The Base Amount shows the total amount derived from the cross-referenced line(s) that the current line then used to calculate its billed amount.
- **Rate Component Sequence** refers to the sequence number of the rate component on the applicable rate version that was used to calculate the line.
- **Measures Peak Qty** is checked if the UOM priced on the calculation line is used to measure a peak quantity.
- **Exempt Amount** is the amount of the calculated charge that the taxpayer doesn't have to pay because they are tax exempt.

- **Distribution Code** is the distribution code associated with the calculation line. This distribution code is used to build the general ledger details on the bill segment's financial transaction.
- **Description** describes the characteristic value that was used when the line's amount was calculated. This information is only displayed if the line was calculated using a rate factor (because only rate factors use characteristic values). Refer to [An Illustration Of A Rate Factor And Its Characteristics](#) for more information.

For more information, refer to [Calculated Adjustments](#).

The **Adjustment Actions** area contains buttons that you use to commit and cancel the adjustment's financial impact. The adjustment's status controls the button you can see and select. Refer to [The Lifecycle Of An Adjustment](#) for the details.

**Cancel may not be enabled.** The cancel button is not enabled if the adjustment is linked to a bill that is written off.

**Canceling An Accounts Payable (A/P) Adjustment.** If you need to cancel an A/P adjustment (i.e. an adjustment of an adjustment type that has an A/P request type) that has already been extracted by A/P, refer to [How To Cancel An A/P Adjustment After It's Been Selected By A/P](#).

## Adjustments - Characteristics

You use this page to link additional information to the adjustment. Open using **Financial, Adjustment, Characteristics**.

### Description of Page

The characteristics collection contains information that describes miscellaneous information about the adjustment.

**Characteristic Types.** You can only choose characteristic types defined on the adjustment's [adjustment type](#).

The following fields display:

<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

**Default Note.** An adjustment's characteristics default from the [adjustment type](#).

## Adjustments - Transfer Adjustment

The Description of Page section that appears below simply describes the fields on this page. Refer to [How and When To Use An Adjustment](#) for instructions describing how to perform common maintenance functions.

The Transfer Adjustment is used to define the reciprocal adjustment associated with a transfer adjustment (there are always two adjustments associated with a transfer). Open using **Financial, Adjustment, Transfer Adjustment**.

### Description of Page

**Adjustment Info** contains a concatenation of the adjustment amount, adjustment type and status.

**Adjustment ID** is the system-assigned, unique identifier of the adjustment.

**Account ID** is the account to which the adjustment is linked. The name of the main person on the account appears next to the account ID.

Use **Obligation ID** to define the transfer to obligation. Basic information about the obligation appears adjacent.

**Location** is a display-only field that shows the location that associated with the obligation, if any.

The **Adjustment Type** and adjustment **Amount** from the first page are displayed. If you need to change either value, return to the first page.

**Adjustment Status** shows the status of the transfer to adjustment. If you need to change the adjustment's status, use the action buttons.

Use the **Comments** to describe anything unusual about the adjustment.

**Transfer Adj ID** is the system-assigned, unique identifier of the adjustment.

The **Creation Date** defines the date on which the adjustment was created.

The financial transaction grid contains the financial transactions associated with the adjustment. It only contains information after the adjustment is frozen. If the adjustment is canceled, a second row appears showing the details of the cancellation FT.

- **Financial Transaction ID** is the system-assigned unique identifier of the FT. Click the go to button to transfer to the financial transaction. On this page you can change certain aspects of the FT in question.
- **Effective Date** is the date the FT starts aging.
- **Accounting Date** is the date the system uses to determine the FT's accounting period in your general ledger.
- **Current Amount** contains the FT's effect on the obligation's current balance.
- **Payoff Amount** contains the FT's effect on the obligation's payoff balance.
- **Bill ID** is the bill on which the FT appears (if it has been swept onto a bill). Click the adjacent go to button to transfer to the bill on which the FT appears. Note: an FT is linked to a bill the next time a bill is completed for the account.
- **Group FT ID** is the identifier of the main tax liability assessment's FT. Refer to [Group Financial Transactions](#) for more information.

## Adjustments - A/P Request

The Description of Page that appears below simply describes the fields on this page. Refer to [How and When To Use An Adjustment](#) for instructions describing how to perform common maintenance functions.

The A/P Request page contains information about adjustments used to refund money via an A/P check request. This page is only relevant if:

- The adjustment's adjustment type references an A/P Request Type (i.e., it will be interfaced to your accounts payable system which will cut the check), AND
- The adjustment is **frozen**.

Open this page using **Financial, Adjustment, A/P Request**.

### Description of Page

**Adjustment Info** contains a concatenation of the adjustment amount, adjustment type and status.

**Adjustment ID** is the system-assigned, unique identifier of the adjustment.

**Account ID** is the account to which the adjustment is linked.

**Adjustment type controls A/P check requests.** The A/P Request information is populated when an adjustment used to refund money via an A/P check request is **frozen**. Whether or not an adjustment is interfaced to A/P is controlled by the adjustment's adjustment type. The adjustment type's A/P Request Type controls the payment date and the bank.

**Name** is the name printed on the check. This name is derived from the account's main person:

- If this person has an override mailing name, the first line of the override mailing name is used.
- If this person does not have an override mailing name, the person's primary name is used.

**Payment Selection Status** is the status of the check request. The values are:

- **Not Selected for Payment** before it's selected by A/P for payment
- **Paid** after it's selected by A/P
- **Canceled** if it's been canceled in A/P
- **Hold** if it's been held in A/P

**Scheduled to Pay** is the date on which the check is scheduled to be cut. This is equal to the adjustment date plus the Due Days on the adjustment type's A/P request type.

The following fields are provided in order to support a two-way interface with an A/P system. The delivered system does not have processes which update these fields.

- **Payment Date** is the date on which the check was cut in A/P. This field is only populated after A/P cuts the check.



- **Paid Amount** is the amount of the check. This field is only populated after A/P cuts the check.
- **A/P Request ID** is the system-assigned, unique identifier of the A/P check request.
- **Payment Number** is the system-assigned number of the payment in A/P (this number typically appears on the printed check). This field is only populated after A/P cuts the check.

The address information in the bottom frame is the address to which the check is mailed. This is the billing address of the main taxpayer linked to the account. After the adjustment is frozen, this information is automatically populated based on the address information for the account. This information is modifiable until the status is ***Paid***.

## Adjustments - Approval

This page only appears after an adjustment has started the approval process. Open this page using **Financial, Adjustment, Approval**.

Refer to [The Big Picture of Adjustment Approvals](#) for more information about the approval process.

The topics in this section describe the base-package zones that appear on the Approval Profile portal.

### Contents

- [Adjustment Information](#)
- [Approval Request](#)
- [Approval Request Log](#)

### Adjustment Information

The Adjustment Information zone contains display-only information about the adjustment.

### Approval Request

The Approval Request zone shows the current and future approvers of an adjustment. This zone only appears if the adjustment is in the approval process.

If the current user has approval authority, the following functions are available:

- Click the **Approve** button to approve the adjustment.
- Click the **Reject** button to reject the adjustment.

### Approval Request Log

The Approval Request Log zone contains the history of the adjustment's approval.

## Financial - Adjustment Calculation Line Characteristics

This page displays the characteristics that are linked to a specific adjustment calculation line. The information on this page is for audit purposes only and cannot be changed.

Open this page by clicking on the characteristics go to button on the calculation lines grid of the [Adjustments - Main](#) page.

### Description of Page

**Account** is the account to which the adjustment is linked. The name of the main person on the account appears next to the account ID.

The **Obligation** is the obligation whose value needs to be adjusted. Basic information about the obligation appears adjacent.

**Location** shows the location that is associated with the obligation, if any.

The **Adjustment Info** displays information to identify the adjustment, including amount, type, and status.

**Adjustment ID** displays the identification of the adjustment, and **Calc Line** displays the calculation line with which the characteristics are associated.

The **Description on Bill** displays the description of the calculation line that appears on the taxpayer's bill.

The characteristics grid displays the **Characteristic Types** and the **Characteristic Values** associated with the calculation line.

For more information about characteristics, refer to [Setting Up Characteristic Types and Their Values](#).

## How and When To Use An Adjustment

The topics in this section describe how to perform common adjustment maintenance functions.

### Contents

- [How To Create A Transfer Adjustment](#)
- [How To Create A Calculated Adjustment](#)
- [How To Cancel An A/P Adjustment After It Has Been Selected By A/P](#)
- [How To Apply Ad Hoc Fees To An Obligation](#)
- [How To Change The Age Of Debt](#)
- [How To Use An Adjustment To Change The GL Distribution](#)

## How To Create A Transfer Adjustment

The following steps describe how to create a transfer adjustment on the Adjustment page.

**Note.** You cannot use an adjustment type with a calculated amount for transfer adjustments.

**Warning!** This process only makes sense in the context of the page used to maintain Adjustments. Refer to [Adjustments - Main Information](#) for the details.

To create a transfer adjustment:

- Specify the "transfer from" account and obligation on the Adjustment - Main page.

- Specify the adjustment type and the amount and transfer to the Transfer Adjustment page.
- Specify the “transfer to” obligation and click **Generate**. In the Adjustment Generate popup page, you can specify the **Effective Date** and **Group FT ID** for the Adjustment FT. Check **FT Header** to indicate if the adjustment FT will be used as the assessment header.
- After generating, click **Freeze** to freeze the adjustments’ financial transactions.

## How To Create A Calculated Adjustment

The following steps describe how to create a [calculated adjustment](#) on the Adjustment page.

**Warning!** This process only makes sense in the context of the page used to maintain Adjustments. Refer to [Adjustments - Main Information](#) for the details.

To create a calculated adjustment:

1. Open **Financial, Adjustment, Main** and specify the **Obligation** to which the adjustment will be applied.
2. Specify the **Adjustment Type** that calculates the amount, and enter a base **Amount**.
3. Click **Generate**. The **Generate** dialog box opens.
4. In the resulting dialog box, specify an **Accounting Date** and **Calculation Date**. You may also specify the **Effective Date** and **Group FT ID** for the Adjustment FT. Check **FT Header** to indicate if the adjustment FT will be used as the assessment header. Click **Calculate**.
5. Click **Freeze** to freeze the calculated transactions.

The **Calculation Date** that you supply is used by the generate adjustment algorithm for any calculations that are effective dated (e.g., rate version or rate factor value).

## How To Cancel An A/P Adjustment After It Has Been Selected By A/P

Adjustments that are interfaced to A/P (because A/P needs to cut a refund check) sometimes need to be canceled. You may cancel an A/P adjustment in the system while its A/P request is **Not Selected**. If the A/P request is **Paid** or on **Hold** you must first cancel the payment in A/P. Canceling the payment in A/P changes the A/P request to **Canceled**. At this point, you can cancel the adjustment.

## How To Apply Ad Hoc Fees To An Obligation

If you need to apply a fee (e.g., NSF charge) to an obligation, issue an adjustment using the appropriate adjustment type. The adjustment amount should be a positive number so that the taxpayer’s balance increases.

## How To Change The Age Of Debt

If an obligation’s debt is older (or younger) than what it should be (for whatever reason), you should do the following.

- Use the Transfer Adjustment process to re-age the debt. The 'transfer from' and 'transfer to' obligations will be the same. The adjustment amount should be a negative number. Refer to [How To Create A Transfer Adjustment](#) for more information.
- After creating the transfer adjustment, drill into the transfer to adjustment and populate the effective date with the day on which the debt should start aging (e.g., if the debt should be 7 days old, the effective date should be the current date minus 7 days).

## How To Use An Adjustment To Change The GL Distribution

Assume you have a frozen bill segment that affected an incorrect GL distribution code. You have two ways to correct such a situation:

- You could correct the cause of the incorrect distribution code (probably incorrect GL information on a rate component) and cancel / rebill the bill segment. Refer to [Cancel / Rebill](#) for more information about this method.
- You could create an adjustment that does not impact the taxpayer's balance but does have a GL entry. This GL entry would reverse the effect of the original distribution code and have an offsetting entry to the correct distribution code. The following points explain how to do this:
  - Choose an adjustment type that only impacts the GL (i.e., the adjustment's adjustment type has an FT algorithm that doesn't impact the taxpayer's current or payoff balance, it only impacts the GL).
  - Enter an adjustment amount of zero.
  - Generate the adjustment.
  - Drill into the adjustment's [financial transaction](#) and enter two (or more) GL distribution rows that reflect the redistribution of the amounts.
  - Return to the adjustment and freeze it.

**Adjusting statistical amounts.** You can use the above approach to change the statistical amount that is interfaced to the general ledger. The statistical amount is simply another column on the financial transaction's GL distribution rows.

## Interfacing Adjustments From External Sources

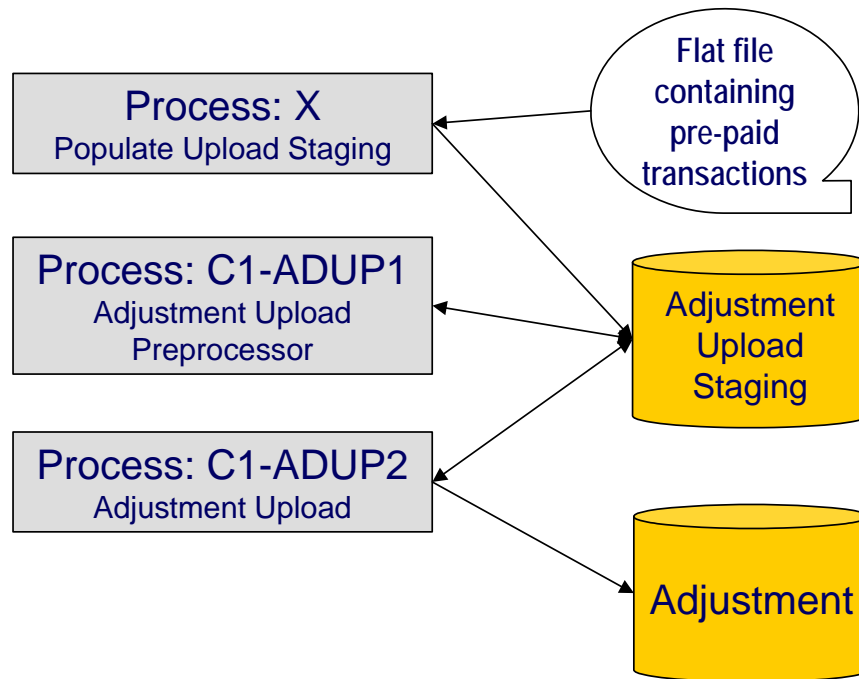
The topics in this section describe how adjustments are uploaded from an external source.

### Contents

- [Interfacing Adjustments](#)
- [Suspense Adjustments](#)
- [Maintaining Adjustment Staging Control](#)
- [Maintaining Adjustment Upload Staging](#)

## Interfacing Adjustments

The following diagram illustrates the processes involved in the uploading of adjustments into the system.



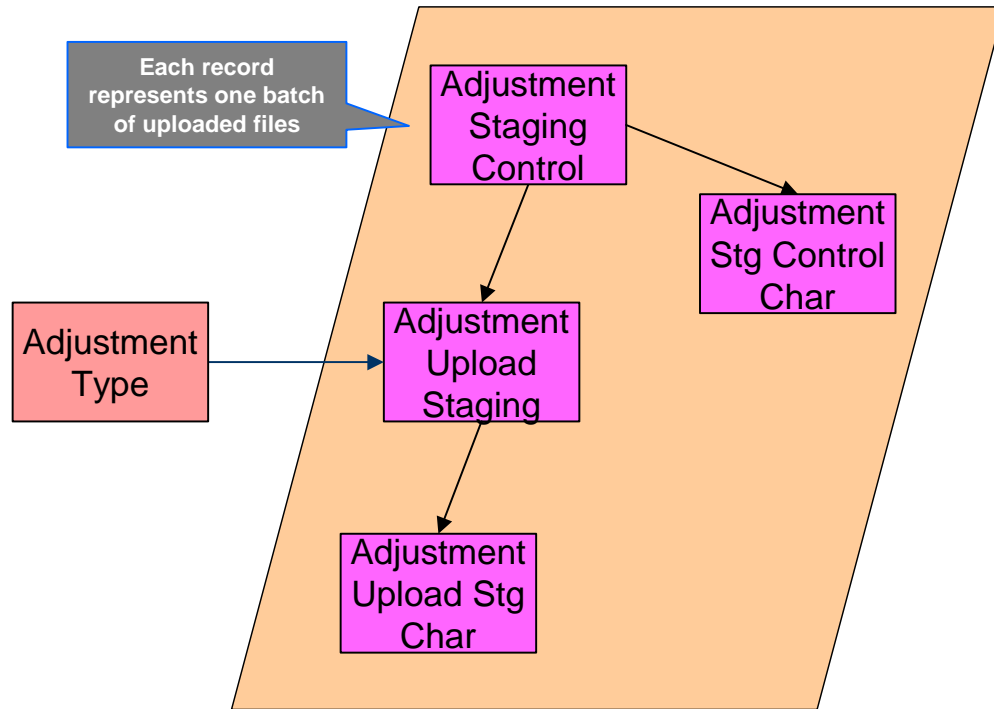
The topics in this section describe how these processes work.

### Contents

- Process X - Populate Adjustment Upload Records
- Process C1-ADUP1 - Preprocess Adjustment Uploads
- Process C1-ADUP2 - Upload Adjustments

### Process X - Populate Adjustment Upload Records

Process X refers to the mechanism used by your organization to populate the various staging tables (shown in the orange section of the following ERD).



The topics in this section describe each of these tables.

### Contents

- [Adjustment Staging Control](#)
- [Adjustment Staging Control Characteristic](#)
- [Adjustment Upload Staging](#)
- [Adjustment Characteristic Upload Staging](#)
- [The Lifecycle of an Adjustment Staging Control Record](#)
- [The Lifecycle of an Adjustment Upload Staging Record](#)

### Adjustment Staging Control

You must create an adjustment staging control record for each batch of adjustments to be uploaded into the system. The name of this table is CI\_ADJ\_STG\_CTL. The following describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
ADJ_STG_CTL_ID	12	Y	N	This is the unique identifier of the adjustment staging control record. This key is a sequential number so you can use a database function to assign the value when populating the table.
CRE_DTTM	15	Y	DateTime	This is the date / time on which the adjustment staging control record was created. This must be populated with the current date / time.
ADJ_STG_CTL_STATUS_	4	Y	A/N	This must be set to <i>P</i> for <i>Pending</i> .

FLG				
ADJ_STG_UP_REC_CNT	10	Y	N	This is the total number of adjustment upload staging records that are linked to this adjustment staging control.
TOT_ADJ_AMT	13.2	Y	N	This column must equal the sum of adjustment amounts on the adjustment upload staging records that are linked to this adjustment staging control.
CURRENCY_CD	3	Y	A/N	This must be a valid currency code in the system. Refer to <a href="#">Defining Currency Codes</a> for more information.
MESSAGE_CATEGORY	5	N	N	Leave this blank. The adjustment upload preprocessor populates this when an error occurs during upload.
MESSAGE_NBR	5	N	N	Leave this blank. The adjustment upload preprocessor populates this when an error occurs during upload.

#### Adjustment Staging Control Characteristic

You must create an adjustment staging control characteristic record for each characteristic that you would like to link to the adjustment staging control. The name of this table is CI\_ADJ\_STG\_CTL\_CHAR. The following describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
ADJ_STG_CTL_ID	12	Y	N	This must correspond with the prime key of the related CI_ADJ_STG_CTL record.
CHAR_TYPE_CD	8	Y	A/N	This must correspond with a characteristic type that is defined as valid for <i>adjustment staging control</i> . Refer to <a href="#">Setting Up Characteristic Types &amp; Their Values</a> for more information.
SEQ_NUM	3	Y	N	This should be set to <b>10</b> unless you have multiple values for a given adjustment staging control and characteristic type.
CHAR_VAL	16	N	A/N	Populate this field if your <a href="#">characteristic type</a> is <i>predefined</i> .
ADHOC_CHAR_VAL	254	N	A/N	Populate this field if your <a href="#">characteristic type</a> is <i>ad-hoc</i> or <i>file location</i> .
CHAR_VAL_FK1 - CHAR_VAL_FK5	50 each	N	A/N	Populate these fields if your <a href="#">characteristic type</a> is <i>foreign key</i>

				<a href="#">reference</a> . Up to five columns of 50 bytes each are provided to accommodate compound keys.
--	--	--	--	--

### Adjustment Upload Staging

You must create an adjustment upload staging record for each adjustment you want to upload. The name of this table is CI\_ADJ\_STG\_UP. The following describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
ADJ_STG_UP_ID	12	Y	N	This is the unique identifier of the adjustment upload staging record. This key is a sequential number so you can use a database function to assign the value when populating the table.
ADJ_STG_CTL_ID	12	Y	N	The ID of the adjustment staging control that is linked to this adjustment upload staging record.
ADJ_TYPE_CD	8	Y	A/N	This must correspond to the prime key of one of your adjustment types. Refer to <a href="#">Setting up Adjustment Types</a> for more information.
ADJ_STG_UP_STATUS_FLG	4	Y	A/N	This must be set to <i>P</i> for <i>Pending</i> .
CREATE_DT	10	Y	Date	The date when the adjustment occurred.
ADJ_AMT	13.2	Y	N	The amount of the adjustment.
ADJ_SUSPENSE_FLG	4	Y	N	This must be set to <i>NSUS</i> for <i>Not In Suspense</i> .
SA_ID	10	N	A/N	This must correspond to a valid obligation in the system.  If you leave this blank and opt to let the system find the obligation during adjustment upload preprocessing, a <i>Determine Obligation</i> algorithm must be plugged in on the associated adjustment type. This plug-in is meant to derive the obligation ID based on supplied miscellaneous information – e.g. information supplied via adjustment characteristic upload staging.
ADJ_ID	10	N	A/N	Leave this blank. The adjustment upload process populates this.
SUSPENSE_ADJ_ID	10	N	A/N	Leave this blank. The adjustment upload process populates this.



MESSAGE_CATEGORY	5	N	N	Leave this blank. The adjustment upload background processes populate this when an error occurs during upload.
MESSAGE_NBR	5	N	N	Leave this blank. The adjustment upload background processes populate this when an error occurs during upload.

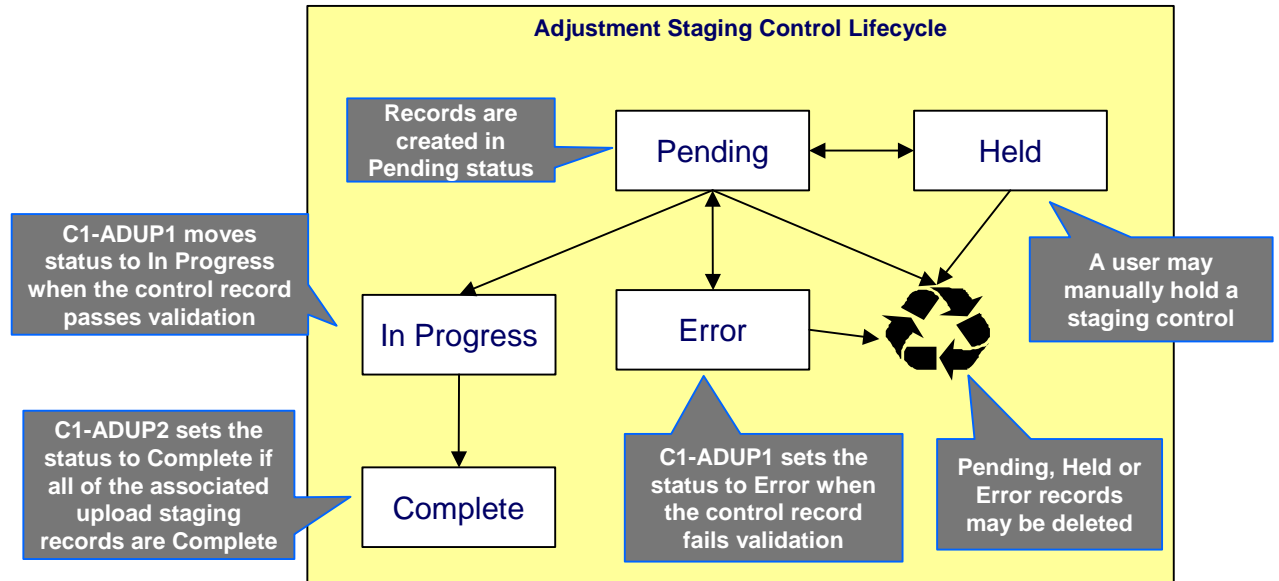
### Adjustment Characteristic Upload Staging

You must create an adjustment characteristic upload staging record for each characteristic that you would like to link to the adjustment upload staging. The name of this table is CI\_ADJ\_STG\_UP\_CHAR. The following describes each column on this table.

Column Name	Length	Req'd	Data Type	Comments
ADJ_STG_UP_ID	12	Y	N	This must correspond with the prime key of the related CI_ADJ_STG_UP record.
CHAR_TYPE_CD	8	Y	A/N	This must correspond with a characteristic type that is defined as valid for <i>adjustment</i> . Refer to <a href="#">Setting Up Characteristic Types &amp; Their Values</a> for more information.
SEQ_NUM	3	Y	N	This should be set to <b>10</b> unless you have multiple values for a given adjustment upload staging and characteristic type.
CHAR_VAL	16	N	A/N	Populate this field if your <a href="#">characteristic type</a> is <i>predefined</i> .
ADHOC_CHAR_VAL	254	N	A/N	Populate this field if your <a href="#">characteristic type</a> is <i>ad-hoc</i> or <i>file location</i> .
CHAR_VAL_FK1 - CHAR_VAL_FK5	50 each	N	A/N	Populate these fields if your <a href="#">characteristic type</a> is <i>foreign key reference</i> . Up to five columns of 50 bytes each are provided to accommodate compound keys.

### The Lifecycle of an Adjustment Staging Control Record

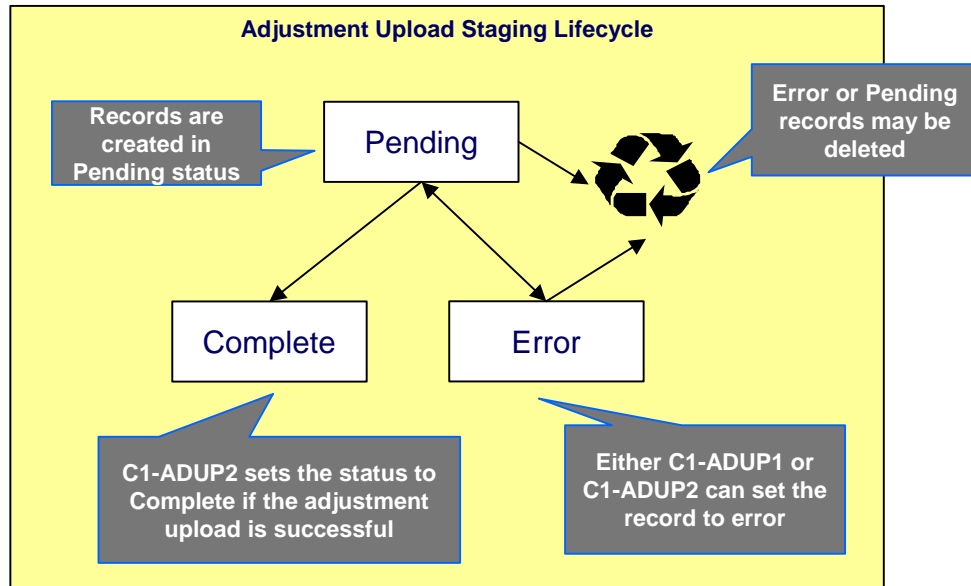
The following diagram shows the possible lifecycle of an adjustment staging control record.



- **Pending.** An adjustment staging control record is created in this state. The **C1-ADUP1** process selects **pending** adjustment staging control records for validation.
- **In Progress.** The **C1-ADUP1** process sets a **pending** record to **in progress** when the totals on the adjustment staging control are successfully validated against the totals from the associated adjustment upload staging records.
- **Complete.** The **C1-ADUP2** process sets the adjustment staging control's status to **complete** when all adjustment upload staging records linked to the adjustment staging control are **complete**.
- **Error.** The **C1-ADUP1** process sets a **pending** record to **error** if the adjustment staging control fails validation. The status may be set back to **pending** after the error is fixed.
- **Held.** The **held** status is available for situations where you want to prevent or delay the upload of a batch of adjustment staging records. The status may be set back to **pending** when the batch of records is ready for upload.
- **Pending, error** and **held** records may be deleted.

#### The Lifecycle of an Adjustment Upload Staging Record

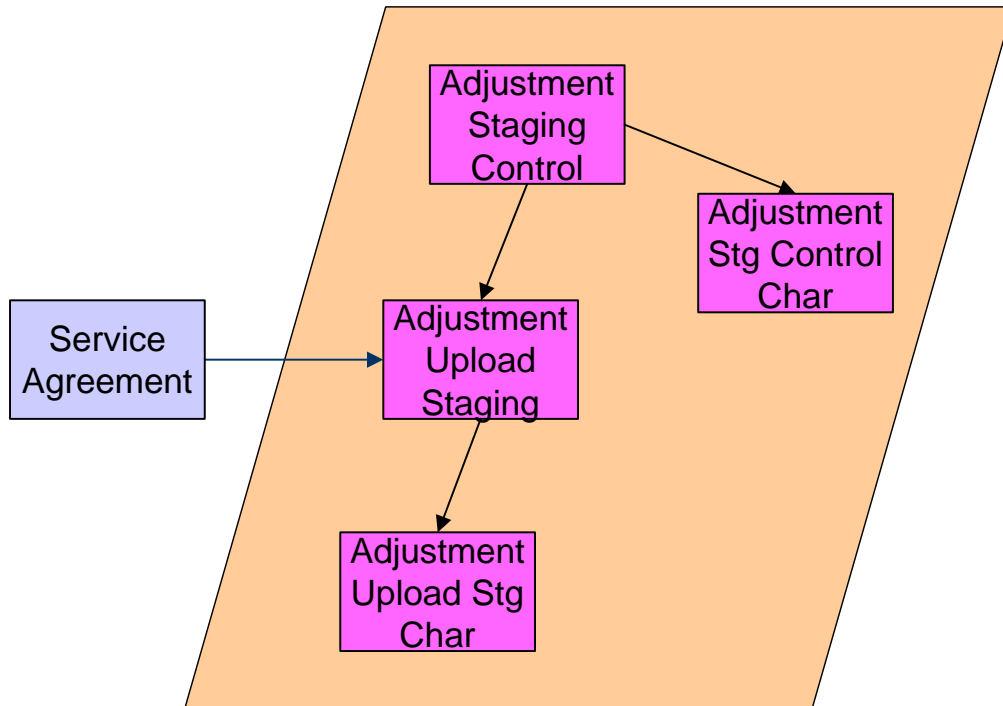
The following diagram shows the possible lifecycle of an adjustment upload staging record.



- **Pending.** Adjustment upload records are created in this state. The **C1-ADUP2** process selects **pending** adjustment upload records and creates adjustments for each of them.
- **Complete.** The **C1-ADUP2** process sets a **pending** record to **complete** if an adjustment is successfully created.
- **Error.** Either **C1-ADUP1** or **C1-ADUP2** process sets **pending** record to **error** when it encounters an error during the upload process. The status may be set back to **pending** after the error is fixed.
- **Pending** and **error** records may be deleted.

### Process C1-ADUP1 - Preprocess Adjustment Uploads

The batch process identified by batch process ID **C1-ADUP1** refers to the background process that validates adjustment staging control records and populates obligation IDs on adjustment upload staging records that do not specify an obligation ID.



## Contents

Phase 1 - Validate Adjustment Staging Controls

Phase 2 - Populate Obligation ID

### Phase 1 - Validate Adjustment Staging Controls

The following points describe, at a high level, the first phase of the adjustment upload pre-process:

- For each **Pending** adjustment staging control,
  - Check that the record count on the adjustment staging control record equals the number of adjustment upload staging records that are linked to the adjustment staging control.
  - Check that the total adjustment amount on the adjustment staging control record equals the sum of the adjustment amounts from the adjustment upload staging records that are linked to the adjustment staging control.
  - If the adjustment staging control passes validation, set its status to **In Progress**. Otherwise, set the status to **Error**. Create a To Do entry using the inputs To Do type and To Do role (if supplied) for adjustment staging control errors. (Complete any outstanding To Do entries for the adjustment staging control before creating a new To Do entry.)
  - If no errors occur, perform To Do cleanup by completing any outstanding To Do entries that were previously created for the adjustment staging control.

**Note.** You can fix errors by going to the To Do entry and drilling into the adjustment staging control page. Don't forget to change the adjustment staging control's status back to **Pending** after fixing the error.

## Phase 2 - Populate Obligation ID

The following points describe, at a high level, the second phase of the adjustment upload pre-process:

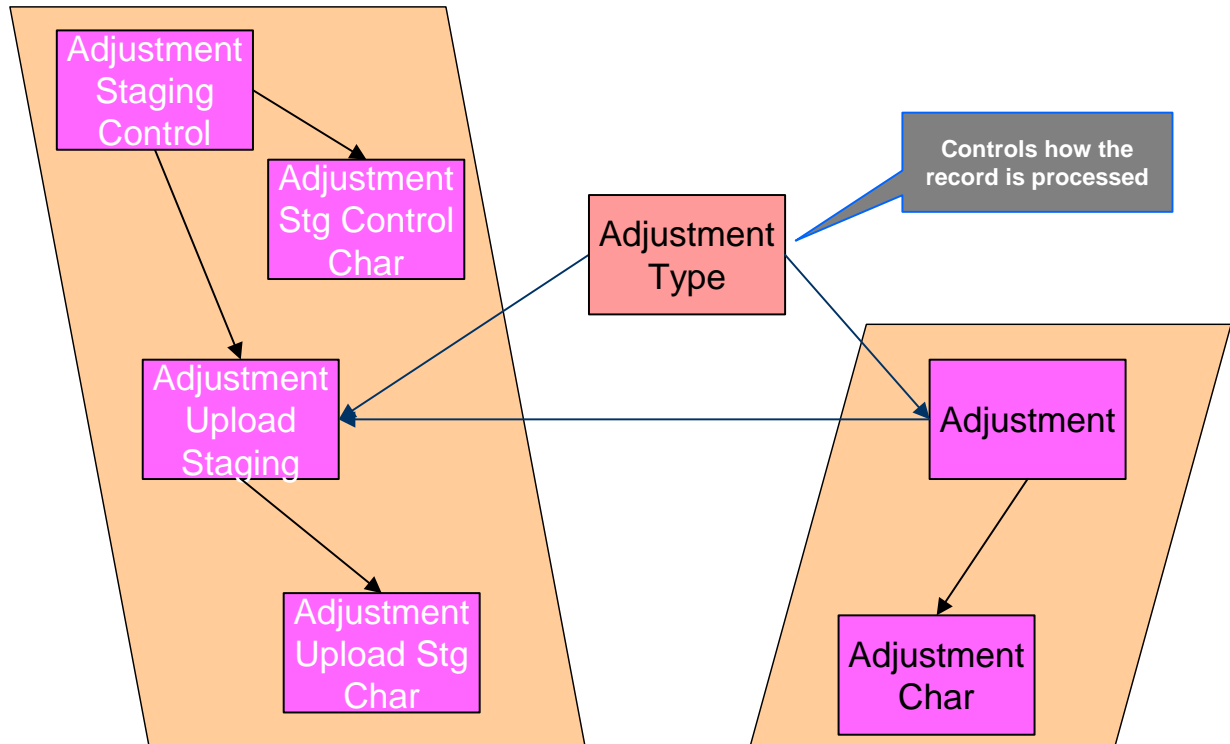
- For each **Pending** adjustment upload staging that is linked to an **In Progress** adjustment staging control AND does not have the obligation ID,
  - Execute the **Determine Obligation** algorithm that is plugged in on the [adjustment type](#).
  - If the algorithm returns a valid obligation ID, stamp that obligation ID onto the adjustment upload staging. If the algorithm also returns an indication that the adjustment is to be put into suspense, set the suspense flag on the adjustment upload staging to **In Suspense**. Refer to [Suspense Adjustments](#) for more information on how suspense adjustments are handled.
  - If the algorithm returns an error, set the adjustment upload staging record's status to **Error**. Create a To Do entry using the inputs To Do type and To Do role (if supplied) for adjustment upload staging errors. (Complete any outstanding To Do entries for the adjustment upload staging before creating a new To Do entry.)
  - If no errors occur, perform To Do cleanup by completing any outstanding To Do entries that were previously created for the adjustment upload staging.

**Note.** You can fix errors by going to the To Do entry and drilling into the adjustment upload staging page. Don't forget to change the adjustment upload staging's status back to **Pending** after fixing the error.

## Process C1-ADUP2 - Upload Adjustments

The batch process identified by batch process ID **C1-ADUP2** refers to the background process that creates adjustments for all adjustment upload staging records that are stamped with an obligation ID.

The following diagram and section describe, at a high level, the processing done in the **C1-ADUP2** background process.



- For each **Pending** adjustment upload staging that has an obligation ID,
  - Add a frozen adjustment using the amount, creation date and adjustment type on the staging record.

**Note.** If the type of adjustment being uploaded is one that is calculated, the algorithm that's plugged in on the adjustment type will calculate the adjustment amount and generate associated calculation lines (if applicable). See [Calculated Adjustments](#) for more information.

- If adjustment creation is successful, update the adjustment ID on the staging record and set the staging record's status to **Complete**.
- If adjustment creation results in an error, set the staging record's status to **Error**. Create a To Do entry using the To Do type on which this batch process (**C1-ADUP2**) is defined as creation process. (Complete any outstanding To Do entries for the adjustment upload staging before creating a new To Do entry.)
- If no errors occur, perform To Do cleanup by completing any outstanding To Do entries that were previously created for the adjustment upload staging.

## Suspense Adjustments

### Contents

[What Are Suspense Adjustments?](#)

[How Are Suspense Adjustments Resolved?](#)

## What Are Suspense Adjustments?

When the adjustment upload preprocessor is unable to identify a valid obligation ID, it can post the adjustment to a suspense obligation. This is similar to how a payment is posted to a suspense account when a valid account ID could not be identified during payment upload.

Putting adjustments in suspense is optional. Therefore, this logic sits in a plug-in spot. If suspense adjustments are applicable to you, you must plug in your suspense logic in a **Determine obligation** algorithm on [Adjustment Type](#).

## How Are Suspense Adjustments Resolved?

A background process can be run periodically to automatically resolve suspense adjustments. Since rules for resolving suspense may vary, this logic sits in a plug-in spot. You must plug in your resolve logic in a **Resolve Suspense** algorithm on [Adjustment Type](#).

### Process C1-ADURS - Resolve Suspense Adjustments

The batch process identified by batch process ID **C1-ADURS** refers to the background process that resolves suspense adjustments.

For each adjustment upload staging that is **In Suspense**, execute the **Resolve Suspense** algorithm that is plugged in on the [adjustment type](#).

## Maintaining Adjustment Staging Control

Use this page to add, view or modify an adjustment staging control record. Open this page using **Financial, Adjustment Staging Control**.

### Description of Page

**Adjustment Staging Control ID** is the unique identifier of the adjustment staging control record.

**Create Date/Time** is when the adjustment staging control was created. This field is protected after the record is saved.

**Number of Staging Records** is the number of adjustment upload staging records that are linked to this adjustment staging control record.

**Total Adjustment Amount** is the total of adjustment amounts from the adjustment upload staging records that are linked to this adjustment staging control record.

Enter the **Currency** for the total adjustment amount on the adjustment staging control. This is also the currency of the adjustment amounts on the associated adjustment upload staging records.

**Status** shows the state of the adjustment staging control. Possible values are: **Pending**, **In Progress**, **Complete**, **Error**, and **Held**.

A **Hold** button appears adjacent to the status description if **Status** is **Pending**. Click this button to set the status to **Held**. The adjustment upload process will ignore staging controls in this state.

A **Pend** button appears adjacent to the status description if **Status** is either **Held** or **Error**. Click this button to set the status to **Pending**. An error message box also appears below the status description if **Status** is **Error**.

Use the **Characteristics** collection to capture miscellaneous information about an adjustment staging control.

<b>Characteristic Type</b>	The type of characteristic.
<b>Characteristics Value</b>	The value of the characteristic.

**Note.** You can only choose characteristic types defined as permissible on an adjustment staging control. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

## Maintaining Adjustment Upload Staging

Use this page to add, view or modify an adjustment upload staging record. Open this page using **Financial, Adjustment Upload Staging**.

### Description of Page

**Adjustment Upload ID** is the unique identifier of the adjustment upload staging record.

**Adjustment Staging Control ID** is the identifier of the adjustment staging control to which the adjustment upload staging is linked.

Indicate the **Adjustment Type**. This field is very important as it controls numerous aspects of the adjustment's impact on the taxpayer's balance and your general ledger.

**You can only choose certain adjustment types.** The obligation's obligation type has a collection of valid adjustment profiles. You may only reference adjustment types that are listed in one of the adjustment type profiles linked to the obligation type.

Enter an **Adjustment Amount**.

**Creation Date** is the date when the adjustment occurred. This will be the creation date on the uploaded adjustment. This field is protected after the record is saved.

**Obligation** indicates the obligation to which the adjustment was posted. If the adjustment is in suspense, the label shows **Suspense Obligation** instead.

**Adjustment** is a reference to the adjustment that got created from the upload.

**Status** shows the state of the adjustment upload staging. Possible values are: **Pending**, **Complete**, and **Error**.

If **Status** is **Error**, a **Pend** button appears adjacent to the status description. In addition, an error message box displays below the status description.

If the adjustment is in suspense, a message appears to indicate the condition and the **Suspense Adjustment** is shown. A different message appears when the suspense is resolved.

Use the **Characteristics** collection to keep miscellaneous information about an adjustment upload staging.

<b>Characteristic Type</b>	The type of characteristic.
<b>Characteristics Value</b>	The value of the characteristic.

**Default Note.** An adjustment upload staging's characteristics default from the [adjustment type](#).



# Financial Transactions

In this section, we describe the financial transactions generated as a result of your bills, payments and adjustments.

## Contents

- [The Big Picture Of Financial Transactions](#)
- [Financial Transaction](#)
- [Account Financial History](#)
- [Account Bill / Payment History](#)
- [Obligation Financial History](#)
- [Obligation Cash Accounting Balance](#)
- [Balance Control](#)
- [Match Event](#)

## The Big Picture Of Financial Transactions

The topics in this section provide background information about a variety of financial transaction issues.

We strongly recommend familiarizing yourself with the topics described in [The Financial Big Picture](#) to fully appreciate the system's financial architecture.

## Contents

- [Bill Segment Financial Transactions](#)
- [Payment Segment Financial Transactions](#)
- [Adjustment Financial Transactions](#)
- [Financial Transactions And Aged Debt](#)
- [Current Balance versus Payoff Balance](#)
- [The Source Of GL Accounts On Financial Transactions](#)
- [Obscure Things That Can Happen](#)
- [The Big Picture of Balance Control](#)
- [The GL Interface](#)

## Bill Segment Financial Transactions

A bill segment has a related financial transaction. The financial transaction contains the financial effects of the bill segment on the obligation's current and payoff balances and on the general ledger.

If a bill segment is cancelled, another financial transaction is created to reverse the original financial transaction. The cancellation financial transaction appears on the next bill produced for the account as a bill correction.

For more information about bill segment financial transactions, refer to [Bill Details](#).

## Payment Segment Financial Transactions

A payment segment has a related financial transaction. The financial transaction contains the financial effects of the payment segment on the obligation's current and payoff balances and on the general ledger.

If a payment segment is cancelled, another financial transaction is created to reverse the original financial transaction. The cancellation financial transaction appears on the next bill produced for the account as a negative payment.

For more information about payment segment financial transactions, refer to [Payment Details](#).

## Adjustment Financial Transactions

An adjustment has a related financial transaction. The financial transaction contains the financial effects of the adjustment on the obligation's debt and on the general ledger.

If the adjustment is eventually cancelled, another financial transaction will be linked to the adjustment to reverse its financial effect. The cancellation financial transaction appears on the next bill produced for the account as an adjustment.

For more information about adjustment financial transactions, refer to [Adjustment Details](#).

## Financial Transactions And Effective Date

When a financial transaction's related bill segment / payment segment / adjustment is frozen, the financial transaction (FT) is also frozen. When an FT is frozen, its obligation's debt is impacted. It is important to stress the following in respect of this impact:

- The FT's GL details will be interfaced to the GL when the GL interface next executes.
- If the FT decreases the amount of debt, the taxpayer's aged debt is affected immediately regardless of whether the FT appears on a bill.
- If the FT increases the amount of debt, the amount the taxpayer owes from an aged debt perspective may or may not be affected by the FT. There is a switch on an FT called New Charge that controls the arrears behavior. If this switch is on, the taxpayer's aged debt will not reflect the FT amount until the FT is swept onto a bill. The moment the FT is swept onto the taxpayer's bill, the debt starts aging. If this switch is off, the date on which the FT starts aging must be defined in the Effective Date field.
- The amount a taxpayer owes in total is immediately affected by the FT regardless of whether the FT appears on a bill. This means that the amount of aged debt may not be in sync with the total amount owed. This seems odd, but is useful from a credit and collections perspective. You see, you probably don't want to start aging a FT until the taxpayer has actually seen it.

## Current Balance versus Payoff Balance

For information about current and payoff balance in general, refer to [Current Amount versus Payoff Amount](#).

The topics in this section describe when payoff amount differs from current amount for the various types of financial transactions.

### Contents

- [Adjustments - Current Balance versus Payoff Balance](#)
- [Billing - Current Balance versus Payoff Balance](#)
- [Payment - Current Balance versus Payoff Balance](#)

### Adjustments - Current Balance versus Payoff Balance

For information about current and payoff balance for adjustments, refer to [Adjustments - Current Balance versus Payoff Balance](#).

### Billing - Current Balance versus Payoff Balance

For information about current and payoff balance for bill segments, refer to [Billing - Current Balance versus Payoff Balance](#).

### Payment - Current Balance versus Payoff Balance

For information about current and payoff balance for payment segments, refer to [Payment - Current Balance versus Payoff Balance](#).

## The Source Of GL Accounts On Financial Transactions

For information about the source of the distribution codes used to generate the GL accounts on the financial transactions, refer to [The Source Of GL Accounts On Financial Transactions](#).

## Obscure Things That Can Happen

The topics in this section provide information about obscure things that can happen when a financial transaction (FT) is frozen.

### Contents

- [A Stopped Obligation May Be Closed](#)
- [A Closed Obligation May Be Reactivated](#)
- [A Reactivated Obligation May Be Closed](#)
- [One Or More Algorithms May Be Executed](#)

## A Stopped Obligation May Be Closed

If an FT causes a **Stopped** obligation's current and payoff balances to become zero, the system closes the obligation (i.e., the obligation's status becomes **Closed**).

## A Closed Obligation May Be Reactivated

After an obligation is closed (i.e., after it's stopped and paid-in-full), it's possible for some types of financial transactions to be linked to the obligation. These financial transactions could cause the current and/or payoff balances to become non-zero. If this happens, the system reactivates the obligation (i.e., the obligation's status becomes **reactivated**).

When an obligation becomes reactivated, it becomes eligible for review by the overdue process monitor (when it next runs).

Financial events that can be linked to a closed obligation are:

- The cancellation of a bill segment
- The freezing of a payment segment
- The cancellation of a payment segment
- The freezing of an adjustment
- The cancellation of an adjustment

## A Reactivated Obligation May Be Closed

After an obligation has been reactivated (refer to the previous section for how this happens), a financial transaction (or transactions) may be linked to it that will cause the current and payoff balance to return to zero. If this happens, the system closes the obligation (i.e., the obligation's status becomes **Closed**).

All types of financial transactions can be posted to a **Reactivated** obligation.

## One Or More Algorithms May Be Executed

If the FT is linked to an obligation whose [obligation type](#) has Freeze Algorithms, these algorithms will be executed when the FT is frozen. In addition, FT freeze algorithms can also be defined on an account's [account type](#).

# The Big Picture of Balance Control

The balance control processes are used to check the financial integrity of your system. The contents of this section describe how these processes work.

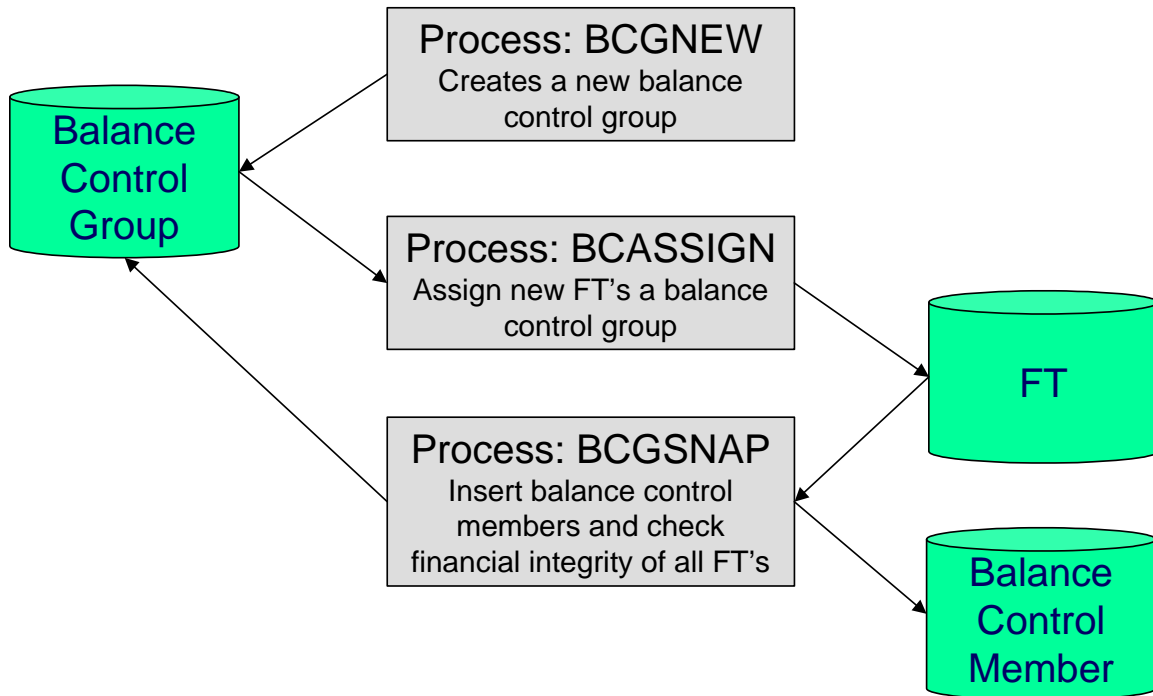
### Contents

[The Balance Control Background Processes](#)

[Balance Control Information Is Available Online](#)

## The Balance Control Background Processes

The following diagram illustrates the balance control background processes:



### Contents

BCGNEW - Create A New Balance Control Group

BCASSIGN - Assign New Financial Transactions A Balance Control Group

BCGSNAP - Insert BC Members And Check Financial Integrity Of All FTs

Consider A Backup At This Point

### BCGNEW - Create A New Balance Control Group

This process creates a **Pending** balance control group if one doesn't already exist. You may wonder why an entire process is dedicated to such a trivial task. The reason is because the next process, BCASSIGN, is a multi-threaded (i.e., parallel) process and we only need one **Pending** balance control group regardless of the number of threads used to assign balance control ID's to financial transactions.

### BCASSIGN - Assign New Financial Transactions A Balance Control Group

This multi-threaded (i.e., parallel) process assigns the **Pending** balance control group to new FTs whose freeze date/time is before the create date/time of the balance control record.

### BCGSNAP - Insert BC Members And Check Financial Integrity Of All FTs

This process performs the following two functions:

- It summarizes new financial transactions under the current Pending balance control group as follows:
  - It creates a balance control member for every combination of **division**, **obligation type** and **FT Type** referenced on the financial transactions belonging to the balance control group.
  - It updates each balance control member with the following information:
    - The number of financial transactions (FTs)

- The sum of the total amounts on the FTs in this balance control group.
- The sum of the current amounts on the FTs in this balance control group.
- The sum of the total amounts from all FTs (in this and all other balance control groups).
- The sum of the current amounts from all FTs (in this and all other balance control groups).
- It sets the status of the balance control group to **Complete**.
- It checks the integrity of the financial transactions in each historical Balance Control Group. It does this by summarizing EVERY financial transaction throughout time and determining if the sums are in sync with the values maintained on the balance control members. If integrity problems are detected, a detailed error message is displayed on the run control associated with the process.

If you opt to run the balance control processes on a nightly basis, you will find that the verification processing will take longer every night (because there are more financial transactions over time). In order to deal with this issue, the [BCGSNAP](#) process has a parameter that allows you to control which of the above functions is implemented (it's call VERIFY-ONLY-SW). In order to speed nightly processing, run this process with the switch set to "G" (this causes new financial transactions to be summarized under a new balance control). Then, once a week (or month), run this process with the switch set to "Y" (this checks the financial integrity of all financial transactions in the system).

If you run the balance control processes less frequently, you can set the VERIFY-ONLY-SW to "N" (this causes both of the above functions to execute).

#### Consider A Backup At This Point

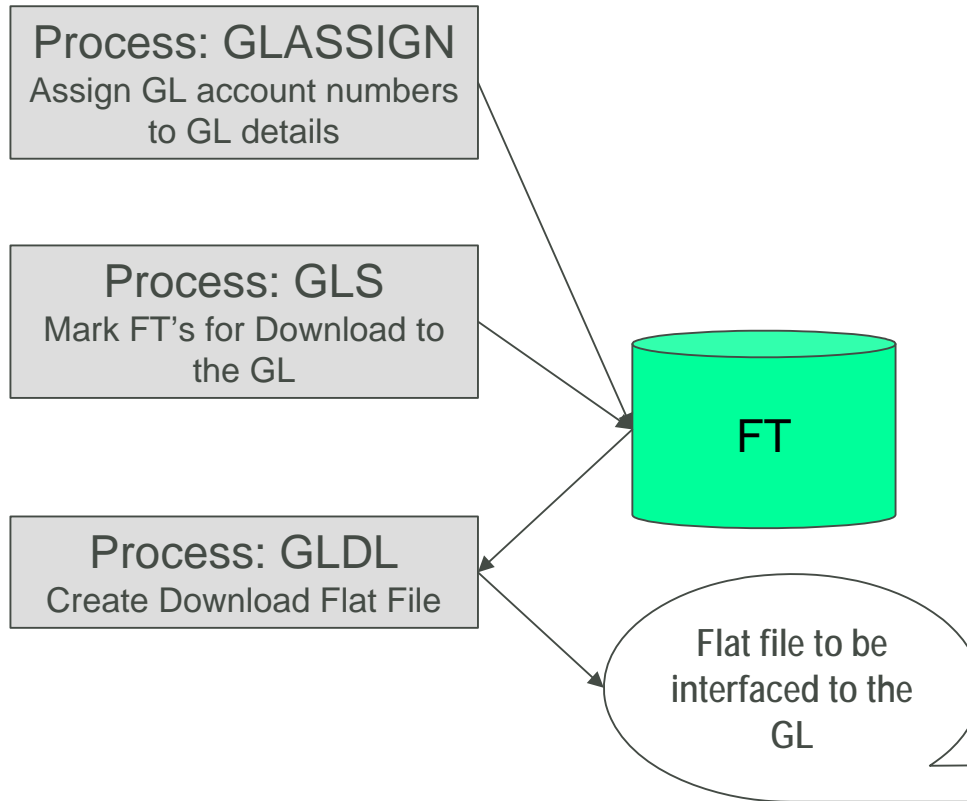
We recommend you backup your database AFTER you run the above processes. Why? So that if a financial integrity problem is spotted in the future, you can compare the current database against the backup to see what changed.

### Balance Control Information Is Available Online

Your internal auditors may be interested in the total number and amount of financial transactions that have been posted since a given point in time. You can use the [Balance Control](#) page to see this information.

## The GL Interface

The following diagram illustrates the GL Interface.

**Contents**

- [GLASSIGN - Assign GL Account Numbers To GL Details](#)
- [GLS - Prepare FTs for Download](#)
- [GLDL - Create General Ledger Download Flat File](#)

**GLASSIGN - Assign GL Account Numbers To GL Details**

The **GLASSIGN** process assigns GL account numbers to the GL details associated with financial transactions. GL account numbers are assigned as follows:

- Every GL detail references a distribution code.
- Every distribution code references a GL assignment algorithm. The base package algorithm simply uses the default GL account associated with the distribution code. However, you can construct your own algorithm(s) to assemble your GL account numbers in your desired fashion. Refer to [Setting Up Distribution Codes](#) for more information about the GL format algorithm.
- The **GLASSIGN** process simply calls each GL's details distribution code's GL assignment algorithm and updates the GL detail with the result (i.e., the GL account number). This GL account number is then used by the **GLDL** process when it creates the consolidated journal entry that's interfaced to the GL.

**If incorrect GL account numbers get assigned to GL details...** If you do not plug in the correct algorithm or your algorithm is wrong, you can correct the GL account numbers that are assigned to the GL details. How? Write a simple program that resets the GL account numbers on the erroneous GL details. Then run **GLASSIGN**. **GLASSIGN** will re-execute the distribution code GL assignment algorithm and refresh the account number accordingly.

## GLS - Prepare FTs for Download

The **GLS** process creates FT download staging records for all FTs that are ready to be posted to the GL (the FT download staging records are stored on the FT/Process table). Each FT download staging record is marked with a batch process ID and run number.

- The batch process ID is the process responsible for creating the flat file that contains the consolidated journal entry that is interfaced to your general ledger. The batch process ID is defined on [Installation Options – Financial](#). The system comes with a single GL download program ([GLDL - Create General Ledger Download Flat File](#)). The GL account numbers are provided by the GL account algorithms that are specified on the distribution codes. Refer to [Setting Up Distribution Codes](#) for more information about the GL account algorithm.
- The run number is the batch process ID's current run number.

This process also changes the status of the FT to **distributed**. You may not change the FT's GL details after this time.

## GLDL - Create General Ledger Download Flat File

The **GLDL** process creates the flat file that contains the consolidated journal entry that is interfaced to your general ledger. One header and multiple detail records are created as described below. At the conclusion of the batch process, a validation is performed to compare debits against credits. If they are not equal, the process terminates with an error.

### Contents

[Header Record Layout](#)

[Detail Record Layout for GLDL](#)

#### Header Record Layout

Field Name	Format	Source/Value/Description
REC_TYPE	A1	1 ( 1 is used for header records )
BATCH_CD	A8	The code for the batch process that created the file. (This value will always be 'GLDL'.)
BATCH_NBR	N10	The batch process will be run many times. This field indicates which of those runs produced a particular output file. This may be thought of as an instance identifier for the batch process.
BATCH_RERUN_NBR	N10	Any given instance of the GLDL batch process may be re-run at any time. If this instance has been re-run, this field will be populated with a number indicating how often.
EXTRACT_DTTM	A26	The date and time of the run that created a particular output file.
DETAIL_REC_CNT	N12	The number of details records
DETAIL_REC_TOTAL_DR	N13.2	This is the sum of FINANCIAL_AMOUNTs from all detail records



Field Name	Format	Source/Value/Description
		that were greater than zero (debits).
DETAIL_REC_TOTAL_CR	N13.2	This is the sum of FINANCIAL_AMOUNTs from all detail records that were less than zero (credits).

### Detail Record Layout for GLDL

**Note.** A record is created for each unique occurrence of REC\_TYPE, GL\_DIVISION, CURRENCY\_CD, GL\_ACCOUNT and ACCT\_PERIOD. If a given GL account has debit and credit FINANCIAL\_AMOUNTs, two records will be created – one shows the debit amount, the other shows the credit amount.

Field Name	Format	Source/Value/Description
REC_TYPE	A1	2 ( 2 is used for detail records )
GL_DIVISION	A5	This is the GL division from the CI_FT record. It determines a) which Accounting Calendar will be used to calculate the Accounting Period (below) and b) the currency of the FT.
CURRENCY_CD	A3	The currency in which the Amount is denominated.
GL_ACCOUNT	A48	Contains the GL account number as supplied by the <a href="#">distribution code's GL account algorithm</a> .
ACCT_PERIOD	A6	An accounting period in the format YYYYPP where YYYY is the fiscal year and PP is the accounting period. This is derived from the GL detail's FT's accounting date and GL division.
FINANCIAL_AMOUNT	N13.2	This is the debit or credit amount. Debits are represented by positive numbers; credits are represented by negative numbers. If a given GL account has debit and credit entries, two records will be created – one shows the debit amount, the other shows the credit amount.
STAT_CODE	A8	This is the GL distribution code's statistic code (if any)
STAT_AMOUNT	N13.2	This is the statistical amount from the GL detail lines

**Note.** The GLASSIGN process updates an FT's GL details with the appropriate GL account number.

## Financial Transaction

Payment segments, adjustments and bill segment have a corresponding financial transaction. The financial transaction is created by the system when any of the source transactions is created. It contains the financial effects of its corresponding adjustment / bill segment / payment segment.

**For background information,** refer to [The Big Picture Of Financial Transactions](#).

The topics in the section describe the financial transaction page.

## Contents

- [Financial Transaction - Main](#)
- [Financial Transaction - FT Process](#)
- [Financial Transaction - Characteristics](#)

## Financial Transaction - Main

Use **Financial, Financial Transaction** to update and maintain financial transaction information.

**Navigating from source transactions.** On the bill, payment, and adjustment maintenance pages, you can click the financial transaction go to button to transfer to this page.

### Description of Page

Most of the attributes on this page are display-only. The following points describe the conditions under which certain fields may be modified:

- **Accounting Date** may be modified until the financial transaction (FT) is interfaced to the GL (i.e., until the **GL Distribution Status** is *Distributed*).
- **New Charge** may be modified unless the FT is linked to a bill. If **New Charge** is turned off, **Effective Date** must be specified.
- **Show on Bill** and **Correction** can be modified until the FT is frozen.

The remainder of this section defines each of the fields on the page.

**FT Type** is the type of financial transaction. Values are: *Adjustment*, *Adjustment Cancellation*, *Bill Segment*, *Bill Segment Cancellation*, *Pay Segment*, and *Pay Segment Cancellation*.

Click the go to button to view the originating transaction on the adjustment, bill, or payment page.

**FT ID** is the system-assigned, unique identifier of the financial transaction.

**Obligation ID** is the system-assigned, unique identifier of the obligation to which the financial transaction is linked.

**Bill ID** is the system-assigned, unique identifier of the bill to which the financial transaction is linked. This field is only populated after the FT is swept onto a bill (and this happens when the next bill is completed for the FT's account).

**Sibling ID** is the system-assigned, unique identifier of the source transaction associated with the FT.

- If this FT is associated with a bill segment, **Sibling ID** contains the unique identifier of the bill segment.
- If this FT is associated with a payment segment, **Sibling ID** contains the unique identifier of the payment segment.
- If this FT is associated with an adjustment, **Sibling ID** contains the unique identifier of the adjustment.

**Parent ID** contains the following:

- If this FT is associated with a bill segment, **Parent ID** contains the unique identifier of the bill with which the bill segment is associated.

- If this FT is associated with a payment segment, **Parent ID** contains the unique identifier of the payment with which the payment segment is associated.
- If this FT is associated with an adjustment, **Parent ID** contains the adjustment type.

**Group FT ID** contains the unique identifier of the FT that represents the assessment header.

**Create Date/Time** are the date and time the FT was created.

**Accounting Date** is the accounting date that will be used by the general ledger to define the accounting period(s) into which the FT will be booked.

**Division** is the division associated with the FT. This comes from the division linked to the FT's obligation's obligation type.

**GL Division** is the GL division associated with the FT. This comes from the GL division linked to the FT's obligation's obligation type.

**Show on Bill** indicates if the FT will be shown on the taxpayer's bill. You should only turn this off for erroneous FTs that should not be shown to the taxpayer. For example, if you cancel / rebill a bill segment and you want to suppress the resultant FTs on the printed bill, turn this switch off.

**New Charge** indicates if the FT's charge only starts aging when the FT is swept onto the next bill produced for the account. If you turn this switch off, you must enter the date the FT starts aging in **Effective Date**.

**Not In Arrears** indicates if the FT's financial impact has been canceled and therefore should not be considered for arrears purposes. This switch is turned on by the system when a FT's source transaction is canceled (both the original FT and the cancellation FT are marked as **Not In Arrears**).

**Match Event ID** is the system-assigned, unique identifier of the match event to which the financial transaction is linked. This field is only enabled for [Open Item Accounts](#). Be aware that changing a financial transaction's match event can result in the balancing / unbalancing of the prior and newly referenced match events.

The **Group FT ID** is the identifier of the main tax liability assessment's FT. Refer to [Group Financial Transactions](#) for more information.

If this is a debit FT, the **Debt Category** identifies how this debit is categorized. If this is a credit FT, the debt category may be populated if the credit is associated with a certain category of debt. Refer to [Debt Categories and their Priorities](#) for more information about how this field is populated.

**Required for base algorithms.** The base P&I calculation algorithm and the base [Determine Detailed Balance](#) algorithm require that every debit financial transaction reference a debt category.

Define a **Debt Category Priority** if this credit financial transaction has a special credit allocation rule that is different from the default rule defined on the obligation type. Refer to [Debt Categories and their Priorities](#) for more information.

**Correction** causes the FT to be summarized in the correction area of the bill-at-a-glance. This is set by the system automatically when a bill segment is cancelled / rebilled after its parent bill is completed (i.e., sent to the taxpayer).

**Redundant** indicates if the FT's financial impact is considered irrelevant. This only happens after an FT has reached an age that is no longer relevant for aging purposes (e.g., when the FT is older than 120 days – or whatever age has been set as the Oldest Bucket Age on the Installation Options) and when its balance is exactly equal to zero with respect to other redundant FTs of the obligation.

**Transferred Out** is not used.

The **Frozen** switch is turned on when the FT has been frozen (i.e., posted to the obligation's payoff and/or current balances). If this switch is on, both **Freeze Date/Time** and **Frozen By** are populated.

**Effective Date** is the date the FT's debt is effective. This field is blank if an FT should not be effective until it is swept onto the account's bill. If you want the FT to be effective immediately or on an historical date (for whatever reason), enter the appropriate **Effective Date**. (Note - you must uncheck the "New Charge" box first in order to do this.)

**Current Amount** contains the FT's impact on the obligation's current amount.

**Payoff Amount** contains the FT's impact on the obligation's payoff amount.

For more information, refer to [Current Balance versus Payoff Balance](#).

**Currency Code** is the currency code associated with the FT's account.

**GL Distribution Status** defines the status of the FT in respect of its interface to the general ledger. When an FT is first created, its status is **Pending**. When an FT is frozen, this value is set to **Generated**. If you modify the GL details, the status becomes **Modified**. After it has been marked for interface to the general ledger by the **GLS** process, the status becomes **Distributed**.

**GL Extract Dates** display the **Scheduled** date that the GL details will be marked for interface to the general ledger. Note that this date is typically only set to a future date for FTs associated with automatic payments. **Actual** displays the date that the GL details were actually posted to the general ledger.

The grid at the bottom shows the FT's debits and credits (i.e., the detail journal lines). Debits are shown as positive amounts, credits are negative amounts. The following points describe rules governing if and how the information in the grid can be modified:

- The FT GL details may not be modified if the FT is **Pending** or **Distributed**.
- All **Amounts** must sum to zero.
- Multiple FT GL details may be set to affect the **Total Amount** (although they are normally generated with one GL checked to affect total amount).
- All FT GL's checked to affect **Total Amount** must sum to the **Payoff Amount** of the FT.

The following information is displayed:

- **Sequence** number is the system-assigned, unique identifier of the journal line.
- **Total Amount** defines if the journal line contains the total of the other journal lines. For example, on a bill segment for property tax, this would be turned on for the receivable account because its amount equals the sum of the other payable and revenue GL accounts.
- **Distribution Code** is the CIS distribution code from which the GL account constituents are derived.

- If the **GL Account** has been populated, the GL account number is displayed.

Refer to [GLASSIGN - Assign GL Account Numbers To GL Details](#) for more information about how the GL account is populated.

- **Amount** defines the journal line's amount.
- **Statistic Amount** defines the statistical amount that will be posted to the GL. This value is only populated on distribution lines created for rate components designated as affecting GL statistical quantity.
- **Characteristic Type** and **Characteristic Value** describe the characteristic value that was used when the line's amount was calculated. This information is only displayed if the journal line was derived from bill calculation lines that were calculated using a rate factor (because only rate factor's use characteristic values). Refer to [An Illustration Of A Rate Factor And Its Characteristics](#) for more information.

**Tax reporting.** It is important to note that a journal line's characteristic value is NOT interfaced to the GL. We have included this attribute on the journal line so that tax reporting can be performed from this system (tax reporting typically necessitates showing each taxing authority - the characteristic value - that participated in a given tax payable GL account).

- If you have configured your installation options to indicate that [fund accounting](#) is **practiced**, the description of the **Fund** associated with this distribution code is displayed.

## Financial Transaction - FT Process

Use **Financial, Financial Transaction, FT Process** to view those ancillary processes that may be triggered as a result of this financial transaction.

**Algorithms cause financial transactions to be linked to batch processes.** Financial transactions get associated with a given process / batch number when certain algorithms are executed. For example, when an external offset financial transaction is generated, it would be linked to a download process for the external division whose debt was collected. The external offset download process would use the linked batch process value to identify all financial transactions that were generated on behalf of the division.

### Description of Page

This page shows the **Batch Processes** associated with a given **Financial Transaction**. Information on this page may not be modified. This information appears purely for audit purposes.

## Financial Transaction - Characteristics

You use this page to link additional information to the financial transaction. Open using **Financial, Financial Transaction, Characteristics**.

### Description of Page

The Characteristics page is a grid containing one row for each characteristic linked to the financial transaction.

**Note.** You can only choose characteristic types defined as permissible on an FT record. Refer to [Setting Up Characteristic Types and Their Values](#) for more information.

#### Description of Page

Select a **Characteristic Type**, **Sequence Number** and **Characteristic Value** to be associated with this Financial Transaction.

## Account Financial History

This page shows how an account's current and payoff balance have changed over time. Use **Financial Query, Account Financial History** to open this page.

#### Description of Page

This page is dedicated to a grid that shows an account's financial events. These events are grouped together by Effective Date, Financial Transaction Type and Parent Id (therefore if there were two payments on the same date, two rows would appear).

**Multiple adjustments of the same type on the same date.** If multiple adjustments with the same adjustment type exist on the same date, their total amount will appear as a single row on this query.

You can use this grid to both view high-level information about these events and to transfer to the respective page in which an object is maintained. The following columns are displayed in the grid:

<b>Effective Date</b>	This is the date the event starts aging. This column will be blank if the FT has not started aging yet.
<b>Financial Transaction Type</b>	This column indicates the type of financial event: <b>Bill Segment</b> , <b>Pay Segment</b> , <b>Bill Segment Cancellation</b> , <b>Pay Segment Cancellation</b> , <b>Adjustment</b> and <b>Adjustment (Cancel)</b> . If the event is related to an adjustment, the adjustment type's description is displayed instead of "Adjustment".
<b>Current Amount</b>	This column shows the financial event's effect on the account's current balance.
<b>Current Balance</b>	This column shows the account's current balance after the financial event.
<b>Payoff Amount</b>	This column shows the financial event's effect on the account's payoff balance. The value is grayed out if it is the same as the current amount.

**Payoff Balance**

This column shows the account's payoff balance after the financial event. The value is grayed out if it is the same as the current balance.

If you need to see more information about a specific financial transaction, click the go to button to transfer to the respective page in which the information is maintained.

**For information** about current and payoff balance, refer to [Current Amount versus Payoff Amount](#).

## Account Bill / Payment History

This page shows an account's bills and payments. Use **Financial Query, Bill / Payment History** to open this page.

**Warning!** For [balance-forward accounts](#), bill rows contain the balance presented on the respective bill, and payment rows contain the amount of the respective payment. However, for [open-item accounts](#), this query behaves differently - see the description of page below for the details.

**Description of Page**

This page is dedicated to a grid that shows the account's bills, payments and payment cancellations. You can use this grid to both view high-level information about these objects and to transfer to the respective page on which an object is maintained.

The area beneath **Account ID** provides you with options that control which transactions appear in the grid. The following points describe the various options:

- Use **Transaction Type Filter** to restrict the type of transactions that appear in the grid. The following options are available:
  - **All**. This option shows all transactions.
  - **Bill**. This option shows all bills.
  - **Not Billed Yet**. This option shows a single line with a summary of frozen financial transactions that have not appeared on a bill yet.
  - **Payment**. This option shows all payments.
  - **Payment Cancellation**. This option shows all payment cancellations.
- Use **Match Event Status Filter** to restrict the transactions based on the status of their [match event](#). This filter only appears if the bill's account is an [open-item](#) taxpayer. The following options are available:
  - **All**. This option shows all transactions regardless of the status of their match events.
  - **Balanced**. This option shows all transactions with at least one **balanced** match event.
  - **Disputed**. This option shows all transactions with at least one **disputed** match event.
  - **Unbalanced**. This option shows all transactions with at least one **unbalanced** match event.

- **Unmatched.** This option shows all transactions with at least one financial transaction that is not linked to a match event.
- Use **Date Range From** and **To** to restrict the transactions based on effective date.

Don't forget to click the search button after changing the filters or after selecting a new Account ID.

For [balance-forward accounts](#), bill rows contain bill information including the balance presented on the respective bill, and payment rows contain the amount of the respective payment.

For [open-item accounts](#), the grid behaves differently:

- The amount on bill rows is equal to the sum of the current charges, adjustments and corrections on the bill. Payment rows contain the amount of the respective payment.
- Each row contains an indication if all of its financial transactions are fully matched.
- A summary of the match status of its financial transactions is shown in the adjacent columns:
  - **Balanced** contains the count and total amount of financial transactions linked to this activity that are linked to **balanced** match events.
  - **Unbalanced** contains the count and total amount of financial transactions linked to this activity that are linked to **unbalanced** match events.
  - **Disputed** contains the count and total amount of financial transactions linked to this activity that are linked to **disputed** match events.
  - **Unmatched** contains the count and total amount of financial transactions linked to this activity that are not linked to any match event.

You can use the hyperlinks to view the detailed financial transactions that are summarized in each cell.

## Obligation Financial History

This page is dedicated to a grid that shows the financial transactions linked to an obligation. Use **Financial Query, Obligation Financial History** to open this page.

### Description of Page

This page is dedicated to a grid that shows an obligation's financial transactions (FT). You can use this grid to both view high level information about these objects and to transfer to the respective page in which an object is maintained.

The following columns are displayed in the grid:

<b>Effective Date</b>	This is the date the FT starts aging. This column will be blank if the FT has not started aging yet.
<b>Financial Transaction Type</b>	This column indicates the type of financial event: <b>Bill Segment</b> , <b>Pay Segment</b> , <b>Bill Cancellation</b> , <b>Pay Cancellation</b> , <b>Adjustment</b> and <b>Adjustment (Cancel)</b> . If the event is related to an adjustment, the adjustment type's description is displayed instead of "Adjustment".



<b>Current Amount</b>	This column shows the FT's effect on the obligation's current balance.
<b>Current Balance</b>	This column shows the obligation's current balance after the financial event.
<b>Payoff Amount</b>	This column shows the FT's effect on the obligation's payoff balance. The value is grayed out if it is the same as the current amount.
<b>Payoff Balance</b>	This column shows the obligation's payoff balance after the financial event. The value is grayed out if it is the same as the current balance.

If you need to see more information about a specific financial transaction, click the go to button to transfer to the respective page in which the information is maintained.

**For information** about current and payoff balance, refer to [Current Amount versus Payoff Amount](#).

## Obligation Cash Accounting Balance

**Only relevant if you practice cash accounting.** This transaction is only relevant if you practice cash accounting (e.g., you only pay the taxing authorities when the taxpayer pays you). Refer to [Payables Cash Accounting](#) for more information about cash accounting.

This page displays a grid that shows an obligation's cash accounting balance for each cash holding distribution code that's been booked to. Use **Financial Query, Obligation Cash Accounting Balance** to open this page.

### Description of Page

The following columns are displayed in the grid:

<b>Cash Accounting Distrib Code</b>	This is a general ledger distribution code used as the holding account for cash accounting. Each holding account that's been used by this obligation will appear here.
<b>Payable Distribution Code</b>	This is the general ledger distribution code to which the payable is (or will be) transferred when the cash event occurs.

**For more information** on how the distribution codes are set up, refer to [Setting Up Distribution Codes](#).

**Payoff Balance**

This column shows the obligation's balance for each cash holding account. If this amount is non-zero, it indicates that for this obligation we are still holding some payables back until payment is received by the taxpayer, at which time some or all of this amount will be transferred to the payable distribution code.

For information about cash accounting, refer to [Payables Cash Accounting](#).

## Balance Control

This page is used to summarize the financial transactions that belong to a particular balance control group.

Refer to [The Big Picture of Balance Control](#) for more information.

**Balance Control information is current only as of the Create Date/Time listed.** The information displayed on the page is captured when the balance control background process is run. Any financial activity subsequent to the create date and time will not be shown.

Use **Financial Query, Balance Control** to open this page.

**Description of Page**

This page displays summarized financial information for a particular balance control group. It also details financial information for all obligation types that belong to the group.

The following fields are displayed on the page:

**Group ID**

This is a sequential value that uniquely identifies a particular balance control run. The Group ID is assigned and incremented automatically every time the Balance Control background process is run.

**Status**

This is the status of the balance control record: **Pending**, and **Complete**. The status of the balance control record will only be pending while the balance control background process is being executed. Balance control information is only reliable if the status is complete.

**Create Date/Time**

This is the date and time that the balance control record was created. Financial information displayed on this page will only contain information for financial transactions frozen before this date and time.

<b>Current Amount</b>	This is the sum of the current amounts of all FTs that belong to the balance control group.
<b>Payoff Amount</b>	This is the sum of the payoff amounts of all FTs that belong to the balance control group.
<b>Current Balance</b>	This is the sum of the current amounts of all FTs that belong to this balance control group plus all earlier balance control groups. Therefore this is the instantaneous current balance of the entire CIS as of the create date and time.
<b>Payoff Balance</b>	This is the sum of the payoff amounts of all FTs that belong to this balance control group plus all earlier balance control groups. Therefore this is the instantaneous payoff balance of the entire CIS as of the create date and time.

The number of FT information details the number of bill segments, pay segments, adjustments and their cancellations that belong to the balance control group.

<b>Adjustment</b>	The total number of adjustments that belong to this balance control group.
<b>Adjustment Cancellation</b>	The total number of adjustment cancellations that belong to this balance control group.
<b>Bill Segment</b>	The total number of bill segments that belong to this balance control group.
<b>Bill Cancellation</b>	The total number of bill cancellations that belong to this balance control group.
<b>Pay Segment</b>	The total number of pay segments that belong to this balance control group.
<b>Pay Cancellation</b>	The total number of pay cancellations that belong to this balance control group.

The **obligation type** scroll summarizes the FTs that belong to a particular obligation type within the balance control group – otherwise all fields are defined as above. You can scroll through all obligation types that belong to the balance control group.

<b>Obligation Type</b>	The obligation type being summarized.
------------------------	---------------------------------------

## Match Event

**Warning!** Match Events are only used if you practice [Open Item Accounting](#).

Match events are used to match debit and credit financial transactions together. When financial transactions are linked to a **balanced** match event, they no longer affect the taxpayer's arrears.

You can use the match event page to do the following:

- Add, change, delete, and cancel match events.
- Add and remove financial transactions from / to a match event.
- Designate financial transactions as being "in dispute" (disputed financial transactions do not affect aged debt until they are resolved).
- View all unmatched financial transactions linked to an account.

Refer to [Match Events](#) for a full description of the lifecycle of a match event and a description of how the system automatically creates match events.

## Contents

[Match Event - Main](#)

[Match Event - FT Details](#)

[Match Event - Subtotals](#)

[How To Perform Common Match Event Functions](#)

## Match Event - Main

A typical match event matches a bill's financial transactions with a payment's financial transactions. The **Main** page allows you to define the bill(s) and payment(s) whose financial transactions are matched together. Use **Financial, Match Event** to open this page.

**Debit must equal credits.** Before a match event impacts a taxpayer's arrearage, its debits and credits must net to zero for every obligation referenced on the match event. Until that time, the financial transactions on a match event continue to affect arrearage. The only exception is the case of [disputes](#).

**You can match any debit and credit.** While the above describes the matching of bills and payments, it's important to remember that a match event matches any type of debit with any type of credit. This means that a match event could match a bill with a credit note, or a payment with a payment cancellation.

**You can match any number of payments under a match event.** While most match events deal with a single bill and a single payment, there's no limitation to the number of payments on a match event. The only restriction is that the debits and credits must net to zero for all obligations.

**It is better not to mix multiple bills on a single match event.** For purposes of bill balance information, it is strongly recommended that you compose your match events with financial transactions limited to a single bill. If you mix financial transactions from multiple bills on a single match event you will not be able to determine the unpaid balance of a partially paid bill.

**You can match specific financial transactions.** While most match events deal with every financial transaction on a bill and payment, a match event can deal with individual financial transactions. For example, a match event could match a bill segment with a combination of a payment segment and a write-off adjustment. If you need to add or remove a specific financial transaction (i.e., a bill segment, payment segment, or adjustment), navigate to the **FT Details** tab. Perhaps a better way to differentiate between this page and the **FT Details** tab is to consider the example of a bill with 100 bill segments. When you link this bill to a match event, you are actually linking its 100 financial transactions. If you wanted to add only a subset of this bill's financial transactions to the match event, you'd use the **FT Details** tab.

Refer to [How To](#) for instructions describing how to perform common match event maintenance functions.

### Description of Page

This page is used to maintain the financial transactions (FTs) that are linked to a match event. The remainder of this section defines each of the fields on the page.

**Match Event Info** and **Match Event ID** only appear after the match event exists on the database. The **ID** is a system assigned random number that stays with the match event for life. **Match Event Info** is a concatenation of important details about the match event and its account.

The next section contains the sum of the **Debit** and **Credit** financial transactions linked to the match event. If the debits and credits do not sum to zero, the **Difference** is also shown.

**Account ID** defines the account whose financial transactions are matched under this match event. This field is gray after the match event is added to the database.

**Match Event Status** defines the state of the match event.

- Match events are initially created in the **open** state. Please note that you may delete an **open** match event.
- The system automatically changes an **open** event's status to **balanced** when the sum of the **Debit(s)** equals the sum of the **Credit(s)** for each obligation on the match event. It's worth stressing that a match event may contain financial transactions from many obligations and each obligation's financial transactions must sum to zero before the match event can become **balanced**.
- You may re**open** a **balanced** event (by adding / removing items so that the match event becomes unbalanced).
- You can cancel a **balanced** or **open** match event by changing the **Match Event Status** to **cancelled**. You must also define a **Cancel Reason** if you cancel a match event. Refer to [How Are Match Events Cancelled?](#) for more information about cancellation.

Turn on **Dispute** if this match event exists to designate certain financial transactions as **disputed**. In addition,

- Describe the reason for the dispute in **Remarks**.
- Link the disputed financial transactions to the match event by selecting the bill(s) below. If only a subset of a bill is disputed:
  - Click on the respective hyperlink in the **Unmatched Debits** column grid (this will transfer you to the next tab with these financial transactions displayed).

- Select the bill segments that you want to designate as disputed.
- Press the **Link / Unlink** button to link the selected bill segments to the match event.

**Disputing items on a balanced event.** The **Dispute** switch will be protected when the match event is **Balanced** or **Cancelled**. If the items on a **balanced** event are being disputed, you must add / remove items so that the match event becomes **Open** before you can turn on the **Dispute** switch.

The remainder of the page is dynamic depending on the **Match Event Status**:

- If the status is **Balanced** or **Open**, a grid appears containing a summary of the bills, payments, and payment cancellations that contribute financial transactions to the match event (note, we refer to these collectively as "contributing objects" in the following discussion). The following columns appear in this grid:
  - **Transaction Type** defines whether the contributing object is a **Bill**, **Payment**, **Payment Cancellation** or **Credit Note**. In the rare situation when unbilled financial transactions are linked to the match event, a transaction type of **Not Billed Yet** appears. If the contributing object is a bill its bill id is also displayed. If sequential bill functionality is enabled, the bill's sequential id is displayed instead.
  - **Matched Activity Information** contains summary information about the contributing object. This column is blank if the transaction type is **Not Billed Yet**.
  - The remaining columns contain the count and amount of each contributing object's financial transactions categorized as follows (note, you can drill down on the count or amount to see the specific financial transactions (FT's) on the next tab).
    - **Matched Debits** summarizes the contributing object's debit FT's that are linked to this match event. A checkbox appears if the count is greater than zero. If you select the checkbox and press the **Link / Unlink** button, these debits will be removed from the match event.
    - **Matched Credits** summarizes the contributing object's credit FT's that are linked to this match event. A checkbox appears if the count is greater than zero. If you select the checkbox and press the **Link / Unlink** button, these credits will be removed from the match event.
    - **Other Debits** summarizes the contributing object's debit FT's that are NOT linked to this match event.
    - **Other Credits** summarizes the contributing object's credit FT's that are NOT linked to this match event.

**Adding more financial transactions to a balanced match event.** When a match event is **Balanced**, the grid showing unmatched FT's is suppressed (and therefore you can't add additional FT's to the match event). To expose this grid, simply change the status of the match event to **Open**.

- If the status is **Open**, a second grid appears containing a summary of the bills, payments, and payment cancellations with at least one unmatched financial transaction (we refer to these collectively as "unmatched objects" in the following discussion). The following columns appear in this grid:

- **Transaction Type** defines whether the unmatched object is a **Bill**, **Payment**, **Payment Cancellation** or **Credit Note**. A transaction type of **Not Billed Yet** is used for unbilled financial transactions (e.g., adjustments generated between bills). If the unmatched object is a bill its bill id is also displayed. If sequential bill functionality is enabled, the bill's sequential id is displayed instead.
- **Unmatched Activity Information** contains summary information about the unmatched object. This column is blank if the transaction type is **Not Billed Yet**.
- The remaining columns contain the count and amount of each unmatched object's financial transactions categorized as follows (note, you can drill down on the count or amount to see the specific financial transactions (FT's) on the next tab).
  - **Unmatched Debits** summarizes the unmatched object's debit FT's that are not linked to any match event. A checkbox appears if the count is greater than zero. If you select the checkbox and press the **Link / Unlink** button, these debits will be added to the match event.
  - **Unmatched Credits** summarizes the unmatched object's credit FT's that are not linked to any match event. A checkbox appears if the count is greater than zero. If you select the checkbox and press the **Link / Unlink** button, these credits will be added to the match event.
  - **Matched Debits** summarizes the unmatched object's debit FT's that are linked to any match event.
  - **Matched Credits** summarizes the unmatched object's credit FT's that are linked to any match event.

The last section contains the running total of the **Debit** and **Credit** financial transactions that have been selected / unselected in the above grids. If debits and credits do not sum to zero, the **Difference** is also shown. These values differ from the values on the top of the page as they are updated when a user selects / unselects an object (whereas the values at the top of the page are only updated when the database is changed).

The **Link / Unlink** button becomes enabled when you select an object in the grids. When you press this button, the financial transactions related to the object are added to / removed from the match event.

## Match Event - FT Details

On the **Main** tab, you can add and remove bills, payments, and payment cancellations to / from a match event. When you do this, you are actually adding and removing all of the financial transactions linked to these objects. For example, when you link a bill with 100 bill segments to a match event, you are actually linking its 100 financial transactions.

You need only use the **FT Details** tab when you need to add or remove specific financial transactions. For example, you would use the **FT Details** tab if you need to remove 1 of a bill's 100 financial transactions from a match event.

Use **Financial**, **Match Event**, **FT Details** to open this page (note, you can also open this page using many of the hyperlinks on the Main and obligation Subtotals tabs).

### Description of Page

This page is used to maintain the financial transactions (FTs) that are linked to a match event. The remainder of this section defines each of the fields on the page.



**Match Event Info** and **Match Event ID** only appear after the match event exists on the database. The **ID** is a system assigned random number that stays with a match event for life. The **Match Event Info** is a concatenation of important details about the match event and its account.

The next section contains the sum of the **Debit** and **Credit** financial transactions linked to the match event. If the debits and credits do not sum to zero, the **Difference** is also shown.

The following **Filters** work together to restrict the financial transactions that appear in the grid. The following points describe the various options (note, don't forget to press the search button after specifying the various filter options):

- Use **Transaction Type Filter** to restrict the type of transactions that appear in the grid. The following options are available:
  - **All**. Use this option if you do not wish to restrict financial transactions based on their transaction type.
  - **Bill**. This option allows you to view a specific bill's financial transactions. When this option is selected, an input field appears in which you identify the **Bill ID**.
  - **Credit Note**. This option allows you to view a specific credit note's financial transactions. When this option is selected, an input field appears in which you identify the **Bill ID** (every credit note has a unique bill ID).
  - **Not Billed Yet**. This option allows you to view financial transactions that haven't appeared on a bill yet (e.g., adjustments and corrections).
  - **Payment**. This option allows you to view a specific payment's financial transactions. When this option is selected, an input field appears in which you identify the **Payment ID**.
  - **Payment Cancellations**. This option allows you to view a specific canceled payment's financial transactions. When this option is selected, an input field appears in which you identify the **Payment ID**.
- Use **Linkage Filter** to restrict the transactions based on the match event to which they are linked. The following options are available:
  - **All**. This option shows all financial transactions regardless of their match event.
  - **Linked to another Match Event**. This option shows financial transactions linked to a different match event.
  - **Linked to any Match Event**. This option shows financial transactions linked to any match event.
  - **Linked to this Match Event**. This option shows financial transactions linked to this match event.
  - **Unmatched**. This option shows financial transactions that are not linked to a match event.

This filter is protected and set to **Linked to this Match Event** if the **Transaction Type Filter** is **All**.

- Use **Debit / Credit Filter** to restrict the transactions based on whether they are debits or credits. The following options are available:
  - **All**. This option shows all debit and credit financial transactions.
  - **Credit**. This option shows all credit financial transactions.
  - **Debit**. This option shows all debit financial transactions.



- Use the **Obligation Filter** to define the types of obligations whose financial transactions appear in the grid. The following options are available:
  - **All**. Use this option if you do not wish to restrict financial transactions based on obligation attributes.
  - **Obligation ID**. Use this option to restrict financial transactions to those linked to a specific **obligation**.
  - **Obligation Type**. Use this option to restrict financial transactions to those whose obligations are linked to a given **division** and **obligation type**.

Don't forget to click the search button after changing the filters.

The **Select All / Unselect All** buttons are used to select financial transactions to add to / remove from the match event (note, after selecting the desired financial transactions, you must also press the **Link / Unlink** button at the bottom of the page).

**50 financial transactions at a time.** Clicking **Select All** selects the first 50 bill segments in the grid. If more than 50 financial transactions exist, you must select them in batches.

The grid that follows contains the financial transactions (FT) that match your search criteria. The following information appears in the grid:

- **Select box.** Select the FT if you want to Link / Unlink it. You implicitly link FT's that are NOT already linked and you implicitly unlink FT's that are already linked. This field is protected if the FT is linked to another match event.
- **FT Amount.** This column contains the amount of the financial transaction.
- **FT Type.** This column indicates the type of financial transaction: **Bill Segment**, **Bill Segment Cancellation**, **Pay Segment**, **Pay Segment Cancellation**, **Adjustment** and **Adjustment Cancellation**.
- **Effective Date.** This column contains the financial transaction's effective date.
- **Obligation Information.** This column contains a summary of the financial transaction's obligation.
- **Remarks.** This column highlights the financial transaction's match status: **Linked to this match event**, **Not linked to a match event**, **Linked to another match event** - match event status.
- **Location Information.** This column contains a summary of the location (if any) associated with the financial transaction's obligation.

The last section contains the running total of the **Debit** and **Credit** financial transactions that have been selected / unselected in the grid. If debits and credits do not sum to zero, the **Difference** is calculated. These values differ from the values on the top of the page as they are updated when a user selects / unselects an object (whereas the values at the top of the page are only updated with the database is changed).

The **Link / Unlink** button becomes enabled when you select a row in the grid. When you press this button, the financial transactions related to the object are added to / removed from the match event.

## Match Event - Subtotals

Before a match event impacts a taxpayer's arrearage, its debits and credits must net to zero for every obligation referenced on the match event. This page shows the sum of the debits and credits for every obligation that contributes at least one financial transaction to the match event. Use **Financial, Match Event, Subtotals** to open this page

### Description of Page

This page shows the sum of the debits and credits for every obligation that contributes at least one financial transaction to the match event.

**Match Event Info** and **Match Event ID** only appear after the match event exists on the database. The **ID** is a system assigned random number that stays with a match event for life. The **Match Event Info** is a concatenation of important details about the match event and its account.

The next section contains the sum of the **Debit** and **Credit** financial transactions linked to the match event. If the debits and credits do not sum to zero, the **Difference** is also shown.

The following **Filters** work together to restrict the obligations that appear in the grid. The following points describe the various options (note, don't forget to press the search button after specifying the various filter options):

- Use the **Obligation Filter** to define the types of obligations whose financial transactions appear in the grid. The following options are available:
  - **All**. Use this option if you do not wish to restrict obligations.
  - **Obligation ID**. Use this option to see a specific **obligation**.
  - **Obligation Type**. Use this option to restrict obligations to those with a given **division** and **obligation type**.
- Use **Status Filter** to restrict the obligations based on whether the sum of the debits and credits they contribute to the match event. The following options are available:
  - **All**. Use this option if you do not wish to restrict obligations based on this status.
  - **Balanced**. This option shows only obligations where the sum of debit and credits nets to zero on this match event.
  - **Unbalanced**. This option shows only obligations where the sum of debit and credits do not net to zero on this match event.

Don't forget to click the search button after changing the filters.

The grid that follows contains the obligations that match your search criteria. The following information appears in the grid:

- **Obligation Information**. This column contains a summary of important information about the obligation.
- **Difference**. This column contains the difference between the **Matched Debits** and **Match Credits**. If this is non-zero, the value appears in red.
- **Matched Debits**. This column contains the sum of debit financial transactions that this obligation contributes to this match event.
- **Matched Credits**. This column contains the sum of credit financial transactions that this obligation contributes to this match event.

- **Location Information.** This column contains a summary of the location (if any) associated with the financial transaction's obligation.

## How To Perform Common Match Event Functions

### Contents

- [How To Find The Match Event Associated With A Financial Transaction](#)
- [How To Dispute An Item](#)
- [How To Match A Small Mismatch](#)

### How To Find The Match Event Associated With A Financial Transaction

If you need to find the match event on which a financial transaction (FT) was matched, display the FT in question on [Financial Transaction - Main](#) and then use the "go to" button adjacent to the **Match Event** to drill to the match event.

**Note.** The easiest way to display a financial transaction is to find its corresponding [bill](#), [payment](#) or [adjustment](#) and then drill down on the desired transaction.

### How To Dispute An Item

Refer to [Disputing Items](#) for background information about disputes.

If a taxpayer wants to dispute an item:

- Create a match event for the account.
- Turn the match event's dispute switch on.
- Link the disputed financial transaction to the match event.
- Describe in the match event's comments the reason for the dispute.

### How To Match A Small Mismatch

Assume the following scenario arises:

- A bill is produced for \$2000
- The taxpayer pays \$1993
- An unbalanced match event will result because the taxpayer didn't pay exactly what is owed

If you want to match this payment to the bill (and leave \$7 for the next bill), do the following:

- Create a transfer adjustment of \$7 where the transfer from / to obligation is the same. This results in a debit financial transaction (FT) of \$7 and a credit FT of \$7.
- Create a match event (or update the unbalanced match event) where the matched FTs are:
  - The \$1993 payment
  - The credit side of the transfer adjustment (\$7)
  - And the \$2000 bill

- Then, if the taxpayer pays their next bill in full, the \$7 debit (associated with the transfer adjustment) will be swept onto it.

**Automating small mismatches.** The algorithm responsible for matching a payment to a specific bill can have a tolerance amount defined on it. If the payment is within the tolerance limit, this algorithm will do the above for you. In other words, you don't have to manually do the above if you populate the tolerance limit appropriately on this algorithm. Refer to [DSOV BILL-ID](#) for more information about this algorithm.

# Overpayment Processing

We use the term **overpayment** to refer to an obligation that has a credit balance as a result of payments exceeding the liability. In this section we describe how to manage the overpayment process.

## Contents

[Background Topics](#)  
[Maintaining Overpayment Processes](#)

## Background Topics

---

The topics in this section provide background information about overpayment processing functionality.

## Contents

[Overpayment Process Type](#)  
[Creating an Overpayment](#)  
[Log Information](#)  
[Approving an Overpayment Process](#)  
[Issues Detected](#)

## Overpayment Process Type

Each overpayment process has an associated overpayment process type. The overpayment process type defines the configuration information that is common to overpayment processes of a given type. Refer to [The Big Picture of Overpayments](#) for a description of the business rules that you control when you set up your overpayment process types.

## Creating an Overpayment

An overpayment process occurs when the [obligation](#) has a credit balance as a result of the payment amount exceeding the liability from the assessment. Another common reason for overpayment is the recalculation of tax, penalty, interest or fee that results in a reduced liability.

An overpayment process can be created in various ways:

- As a result of posting a return. This is the most common scenario. The tax form is processed and results in a creation of an [assessment](#). The calculated tax due is evaluated against the total payments made to determine if there is an overpayment.
- As a result of the balance of the obligation being reviewed by the account monitor.
- Some exceptional cases make it necessary to manually create an overpayment process. For example, if the taxpayer believes he/she is owed tax in multiple jurisdictions and makes a first quarter estimated payments where no tax is actually due. The taxpayer may ask for the payment to be refunded prior to filing a zero tax due return after the fourth quarter.

Regardless of how the overpayment is created, the overpayment process is always associated with a specific obligation.

## Log Information

The overpayment log contains an entry for every recorded event during the lifecycle of the overpayment process. There are two types of log entries:

- **Automatic entries.** The system automatically creates an entry in the log when the overpayment process is created or there is a status change. Users cannot modify or delete these log entries.
- **Implementation specific entries.** In addition to entries created when an overpayment process is created and changes status, your implementation can choose to add specific log entries to highlight specific activities that occur as a result of the overpayment. For example, a log entry can be created when a refund is issued.
- **Manual entries.** Users can add manual entries to record significant events at their discretion.

## Approving an Overpayment Process

Some overpayment processes will require approval by one or more users. You can configure the system to inform users of overpayment process that require approval by creating a to do. Refer to [creating to do entries for approval](#) for the details.

From the to do the user can navigate to the overpayment process portal and use the actions on the [actions zone](#) to either approve or reject the overpayment process.

## Issues Detected

An overpayment process will typically go through a number of automated checks or validations before it is processed. If any of the validations fail, the overpayment process will transition to an issues detected or error state.

An issues list will appear at the top of the overpayment process zone. The user can edit the overpayment process and correct any errors. If the issue is related to a specific field, a 'go to' button will appear next to the issue. After saving the changes, the user can then use the **Validate** action for the system to apply the validation rules on the updated overpayment process.

## Maintaining Overpayment Processes

---

Use the Overpayment Process transaction to view and maintain pending or historic overpayment processes. Navigate using **Main Menu, Financial, Overpayment Process**.

### Contents

- [Overpayment Process Query](#)
- [Overpayment Process Portal](#)

## Overpayment Process Query

Use the [query portal](#) to search for an overpayment process. Once a request is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Overpayment Process Portal

This portal appears when an overpayment process has been selected from the Overpayment Process Query portal.

The topics in this section describe the base-package zones that appear on this portal.

### Contents

- [Overpayment Process Actions](#)
- [Overpayment Approval](#)
- [Overpayment Process](#)
- [Overpayment Process Log](#)

### Overpayment Process Actions

This is a standard [actions zone](#).

If the overpayment process is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

In addition to possible next states, an overpayment process would typically include special action buttons to handle additional processing. For example, if the overpayment process is in a state that requires approval, then the **Approve** and **Reject** action buttons would appear.

If the overpayment process is in a final state, the state transition and action buttons are not displayed.

### Overpayment Approval

The Overpayment Approval zone contains display-only information about the current approver of the selected Overpayment Process.

Please see the zone's help text for information about this zone's fields.

### Overpayment Process

The Overpayment Process zone contains display-only information about the selected Overpayment Process.

Please see the zone's help text for information about this zone's fields.

### Overpayment Process Log

This is a standard [log zone](#).

# Rates

Rates may be used to do any number of calculations for a taxpayer, such as assessments for filing based tax types, real property taxes, complicated penalty and interest rules or interest calculations for overpayments.

The rate controls:

- How charges are calculated.
- How the charges appear aesthetically on the taxpayer's bills, for billed obligation types.
- How the general ledger is affected by the charges.

In this section, we explain how to set up your organization's rates.

Refer to [How Rates Affect The Information On A Bill](#) for how billing uses the information in a rate to calculate a bill segment.

**Not all obligations reference a rate.** It's important to understand that the system may not determine the applicable rate for a calculation solely from the obligation. For example, the rate used to calculate an assessment may be referenced on the tax form type. Whether or not an obligation uses a rate and the list of rates that can be specified on an obligation are controlled by the obligation's [obligation type](#).

## Contents

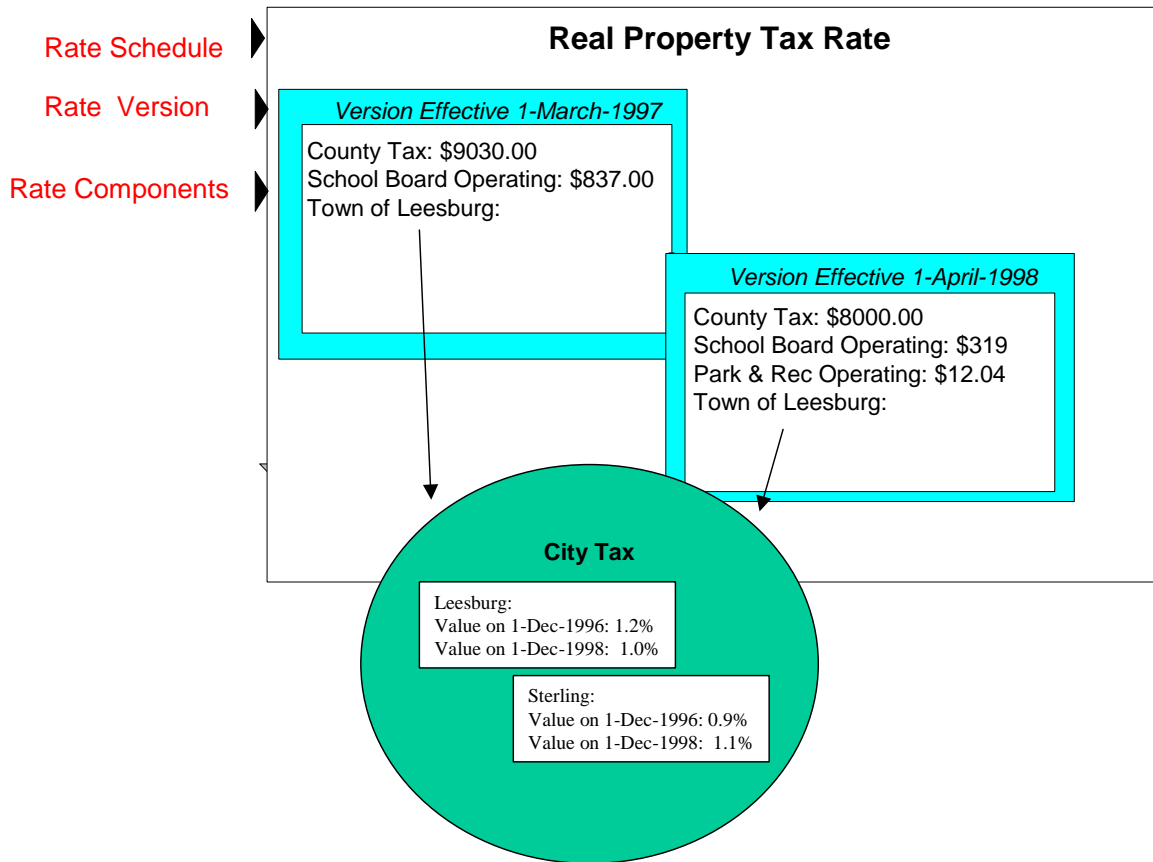
- [All Rates Share A Common Structure](#)
- [How To Create A New Rate](#)
- [Control Tables That Must Be Set Up Before Creating A Rate Schedule](#)
- [Setting Up A Rate Schedule](#)
- [Rate Schedule Merge](#)
- [Setting Up Rate Factors](#)
- [Defining Rate Versions](#)
- [Designing Rate Components](#)
- [Defining Rate Components](#)
- [Rate Version Merge](#)
- [Rate Check](#)
- [Effective Dates & Proration](#)
- [How To Add A New Rate Quantity Rule Algorithm](#)

## All Rates Share A Common Structure

---

All rates share a common structure as illustrated in the following sample rate:





This rate is constructed using the following components:

- Every rate has a single **rate schedule** that contains information about the rate that doesn't change over time. For example, the rate's description.
- A rate's effective-dated calculation algorithms are stored in a **rate version**. Every rate schedule has at least one rate version. Multiple rate versions exist when something about the algorithms changes and it's important to keep the prior version in order to recalculate historical bills. The rate shown above has 2 rate versions; one effective on 1-March-1997, the other effective on 1-April-1998.
- A rate version's calculation algorithms are stored in **rate components**. Every rate version will have at least one rate component. The number of rate components linked to a rate version is dependent on the complexity of the calculation rules. We have seen very simple calculation rules that need only a single rate component; we have seen others that require over 150 rate components. You will find the construction of rate components to be the most challenging task of the setup process.
- **Rate factors** are used to specify the amount to charge when the amount is the same for many rates. Notice that both of the above rate versions reference the prevailing "state tax" rate factor rather than specifying the specific tax percent. This allows the tax rate to be maintained in one place.

**Rate factors have many uses.** Rate factors have many uses in addition to specifying tax rates. Refer to [An Overview Of Rate Factors](#) for all the details.

The remainder of this chapter discusses the intricacies of setting up the rates and rate factors.

## How To Create A New Rate

---

The following points describe steps you'll follow to set up a rate.

- Create a [rate schedule](#) for the rate. Refer to [Setting Up A Rate Schedule](#) for more information.
- Set up a [rate version](#) to establish the earliest effective date of the rate. Refer to [Defining Rate Versions](#) for more information. We recommend that you set the status of the rate version to **In Progress** while you're working on the rate.
- Set up rate components to define the rate version's calculation rules. Refer to [Defining Rate Components](#) and [Rate Version Merge](#) for more information.
- After all rate components have been setup, return to [Rate Version - Main](#) and change the rate version's status to **Validated**. At this point, you can use [Rate Check](#) to test the rate version.
- After you have checked the rate, return to the [Rate Version - Main](#) and change the status to **Finished**. This allows the rate to be used by billing.
- Define the various obligation types that can use the rate. You do this by referencing the rate on [Obligation Type – Rate](#).

## Control Tables That Must Be Set Up Before Creating A Rate Schedule

---

This section describes tables that must be set up before you can add a new rate schedule.

### Contents

- [Defining Frequency Codes](#)
- [Setting Up Rate Quantity \(RQ\) Rules](#)
- [Defining Rate Quantity \(RQ\) Identifiers](#)
- [Setting Up Unit Of Measure Codes](#)
- [Setting Up Time-Of-Use Codes](#)

## Defining Frequency Codes

When you create a rate schedule, you must enter a code defining the frequency in which the charges are expressed, e.g., monthly, quarterly, biannual, etc. To set up frequency codes, open **Admin Menu, Frequency**.

### Description of Page

Enter a unique **Frequency ID** and **Description** for each frequency.

Before entering the remaining fields, you must analyze the 'proration provisions' in your rates. For example, consider a rate used to calculate a tax exemption that is typically granted for bi-annual periods. Assume this rate's amount will not be prorated when the taxpayer qualifies for the exemption between 20 and 160 days. This rate requires a frequency that indicates:

- **2 Periods / Year** (because it's a bi-annual rate).
- **Offset for Minimum Days** equal to **160**.
- **Offset for Maximum Days** equal to **-20**.

When the system creates a bill using a rate with this frequency, it compares the number of days that the taxpayer qualified for the exemption to the minimum and maximum number of days inferred by the rate's frequency. In the example above, the minimum number of days would be 20 (365 days / 2 period per year - 160 offset for minimum days). The maximum number of days would be 163 (365 days / 2 periods per year + (-20) offset for maximum days). Therefore, if a taxpayer qualifies for the exemption for less than 20 days or more than 163 days, the exemption's value is prorated. In this case, prorating the exemption will give the taxpayer some of the benefit while not prorating will either give the taxpayer no exemption or the full exemption value.

**Overriding the system's standard proration logic.** If the standard proration logic does not satisfy your requirements, you may plug in an override proration algorithm to calculate the proration factors as required by your implementation. For example, we have encountered companies who treat a year as having 360 days rather than 365 days. Such a company would need to develop an override proration algorithm and plug it on the [Installation](#) record.

### Where Used

Every Rate Schedule must have a frequency. Refer to [Rate Schedule – Main](#) for more information.

## Setting Up Rate Quantity (RQ) Rules

To set up Rate Quantity (RQ) rules, choose **Admin Menu, Rate Quantity Rule**.

### Description of Page

Enter a unique **RQ Rule ID** and a **Description** for the RQ rule.

Enter a **Long Description** that describes, in detail, what the rule does.

Indicate the **UOM**, **TOU** and **RQI** to be output by the rule.

**Note:** Unit of Measure (UOM) and Time of Use (TOU) do not normally apply to tax-based calculations and will typically be left blank.

Every rate quantity supplied to a rate has two quantities – the quantity as it was initially captured, and the quantity that rate calculation will use. Most of the time, these two amounts will be the same. They will only differ when another RQ Rule has executed and caused the billable quantity to change (an RQ Rule NEVER changes initially captured quantities). Because these two amounts may differ, an RQ Rule needs to know if it should use the quantity as initially captured OR if it should use the quantity that rate calculation will use. You use the **RQ Quantity to Use** field to indicate such. Valid values are *Use Initial Quantity* and *Use Billable Quantity*.

Indicate whether the system should **Create Billing Error** if the RQ Rule is not able to execute successfully. For most RQ Rules you would turn this option on. However, if the rule is linked to a rate where the rule is only applicable to a subset of the taxpayer base using the rate, you may not want a billing error to be issued if the rule cannot execute.

Use **RQ Rule Processing** to indicate if the RQ rule is *Always* executed by rate application, even if the billable details on the original bill segment are to be used when the system recalculates a bill segment (refer to [Rebill \(Bill Segment\)](#)). If the RQ rule should not be executed when the billable details on the original bill segment are used, indicate that the RQ rule should only be executed on the *Initial RQ Calculation*. RQ rules that create a characteristic type / value (in the characteristic collection) used by one or more rate factors referenced in a rate component should typically be always executed. Refer to [Base Package RQ Rules](#) for examples of RQ rules that create a characteristic type / value.

Select the **RQ Algorithm Type** that controls how the system manipulates the quantity supplied to the rate. Define the **Parameters** to be used by the RQ algorithm.

### Base Package RQ Rules

The system provides a number of base package RQ rule algorithms. Click [here](#) to see algorithm types available for this entity.

**Note.** If the base package algorithms are not sufficient for your rating requirements, you must write a new RQ rule algorithm. Refer to [How To Add A New RQ Rule](#) for how to do this.

For more information about defining algorithm types, refer to [Setting Up Algorithm Types](#).

### Where Used

A Rate Schedule may have zero or more RQ Rules. Refer to [Rate Schedule - RQ Rules](#) for more information.

## Defining Rate Quantity (RQ) Identifiers

**Warning!** RQIs exist to support calculations performed by rate components. This means that you must design your rate components before you can design your rate quantity identifiers. We strongly recommend that you design “on paper” how every rate’s rate component looks before you attempt to set up your RQIs.

**Note.** If you create an RQI, you must also have a corresponding RQ Rule to calculate the amount of the RQI to be charged by the rate. If not, it is impossible for the system to generate a bill line for the RQI because there will be no quantity associated with the RQI.

To define rate quantity identifiers, open **Admin Menu, Rate Quantity Identifier**.

### Description of Page

Enter a unique **Rate Qty. Identifier** and **Description** for every RQI.

Enter the appropriate number of **Decimal Positions** for each RQI. The rate application process uses this information to round the calculated rate quantity prior to applying appropriate rate components.

### Where Used

An RQ Rule may use an RQ Identifier to describe what it generates. Refer to [Setting Up Rate Quantity \(RQ\) Rules](#) for more information.

A Rate Quantity Rate Component may use an RQ Identifier to describe what it is charging. In this situation, the rate must contain an RQ Rule that generates a quantity for this RQI; otherwise there will be a charge without a rate quantity. Refer to [Rate Component – Main Information](#) for more information.

## Setting Up Unit Of Measure Codes

Many rates contain prices in respect of some number of something. We use three distinct codes to identify the “thing” being priced in a rate. Unit of measure (UOM) is one code provided to help identify something being priced in a rate.

To define unit of measure codes, open **Admin Menu, Unit of Measure**.

### Description of Page

The following fields display for each unit of measure:

<b>UOM</b>	The unique identifier of the unit of measure.
------------	---

<b>Description</b>	The full description of the UOM.
<b>Tax Type</b>	The type of rate associated with this UOM. Refer to <a href="#">Setting Up Tax Types</a> for more information.
<b>Decimal Positions</b>	The number of decimal positions that appear on bill lines.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI UOM](#).

## Setting Up Time-Of-Use Codes

Many rates contain prices in respect of some number of something. We use three distinct codes to identify the “thing” being priced in a rate. Time of Use (TOU) is one code provided to help identify something being priced in a rate.

To define time-of-use codes, open **Admin Menu, Time of Use**.

**Description of Page**

Enter a unique **Time of Use** code and **Description** for every time of use code.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI TOU](#).

## Setting Up A Rate Schedule

A rate schedule is setup for every rate used in calculations. Rate schedules are defined using the pages described in this section.

**A rate schedule isn't complete without rate versions and rate components.** Refer to [All Rates Share A Common Structure](#) for information about how you must also create rate versions and rate components for every rate schedule.

**Contents**

- [Rate Schedule - Main](#)
- [Rate Schedule - RQ Rules](#)
- [Rate Schedule - Bill Messages](#)

## Rate Schedule - Main

You start the rate definition process by selecting **Main Menu, Rates, Rate Schedule**. On this window you enter general information about the rate.

### Description of Page

Enter a unique **Rate Schedule** ID to identify the rate. The value that you supply will typically correspond with the identity of the rate in your rate book.

Enter a **Description** for the rate schedule.

Select the **Tax Type** that describes the type of tax associated with this rate. When you maintain an existing rate schedule, this field will be gray if the rate is referenced on at least one obligation type. See [Setting Up Tax Types](#) for more information.

Select a **Frequency** to define the time period in which the rate's charges are expressed. See [Defining Frequency Codes](#) for more information.

Select a **Currency Code** to define the currency in which the rate's charges are expressed. See [Defining Currency Codes](#) for more information.

**Currency note.** All rate factors that are referenced on the rate's rate components must be denominated in the same currency as defined on the rate schedule. This is because rate factors are used when the charge is "soft" and all charges in a rate, be they hard or soft, must be in the same currency. Also note – all rates referenced on an account's obligations must be denominated in the same currency as defined on the account.

Turn on **Allow RVs Proration** if a change in the Rate Version during a bill period should be prorated. If changes should not be prorated, indicate the **RV Selection Date** to use when the system determines which rate version to use; available options are:

<i>Accounting Date</i>	The rate version effective on the bill's accounting date will be used.
<i>Bill End Date</i>	The rate version effective on the bill's end date will be used.
<i>Bill Start Date</i>	The rate version effective on the bill's start date will be used.

### Rate Schedule Tree

The right half of this page is dedicated to a tree that shows the rate schedule's RQ Rules and Rate Versions. You can use this tree to view high-level information about these objects. You may also transfer to a given rate version by selecting that rate version.

**Note.** The duplicate action in the action button bar enables you to copy another rate schedule and optionally, its rate versions. Refer to [Duplicate Button](#) in the system wide standards document for more information.

## Rate Schedule - RQ Rules

When a rate schedule has RQ Rules, the system executes each rule's algorithm before it calculates the charges embodied in the rate's rate components. Open **Rates, Rate Schedule** and navigate to the **RQ Rule** to link RQ Rules to a rate.

For more information, refer to [Setting Up Rate Quantity \(RQ\) Rules](#).

**Note.** The [Rate Schedule Merge](#) page enables you to build the RQ rule collection for your rate schedule by copying collections from existing rate schedules.

### Description of Page

Enter a **Rate Quantity Rule** for every type of rate quantity manipulation required by the rate. The **Sequence** number is important when you have multiple rules linked to the rate, as it defines the order in which the rules will be executed.

### Where Used

Billing uses this information to create additional rate quantities at billing time.

## Rate Schedule - Bill Messages

When a rate schedule has bill messages, the system will sweep these messages onto bills that use the rate.

For more information about bill messages, refer to [The Source Of Bill Messages](#).

**Note.** The [Rate Schedule Merge](#) page enables you to build the bill message collection for your rate schedule by copying collections from existing rate schedules.

Select **Main Menu, Rates, Rate Schedule** and navigate to the **Bill Messages** to link bill messages to a rate.

### Description of Page



Use the **Bill Messages** collection to define **Bill Message** codes that should appear on bills that use a given rate schedule. For each message, also specify the **Start Date** and **End Date** when such a message should appear on the bill (leave **End Date** blank if the message should appear indefinitely).

#### Where Used

The system snaps bill messages on a bill during bill completion. Refer to [The Source Of Bill Messages](#) for more information.

## Rate Schedule Merge

---

Use this page to modify an existing rate schedule by copying information from other rate schedules. This page may be used to copy records from the RQ rule, and bill message collections from one or more existing rate schedules to another. Some examples of when this page may be used are as follows:

- You wish to create a new rate schedule, which is similar to an existing rate schedule. Rather than copying all the information from the existing rate schedule and then removing the inapplicable components, this page may be used to selectively copy only the information applicable to the new rate schedule.
- Perhaps you have a tax calculation that contains many optional components. Rather than building a generic rate schedule used by many tax form types, and using other logic to determine which options are applicable to which tax form, you may choose to build a custom rate schedule for each form type, using only the components applicable to that tax form. In this scenario, you may choose to create special 'mini' rate schedules, one for each of the various calculation options. Then, you could use the rate schedule merge page to select the components applicable for the new custom rate schedule.
- Perhaps you are adding several new bill messages, which are applicable to multiple existing rate schedules. Once you have added the new messages to one rate schedule, you may find it easier to update the subsequent rate schedules by using the rate merge page to copy the messages across.

**Note.** The target rate schedule must exist prior to using this page. If you are creating a new rate, you must first go to the [Rate Schedule](#) page to add the new rate schedule and then navigate to the merge page to copy collection information.

### Rate Schedule – Bakery-1

Freq: **Monthly**    Currency: **USD**

RQ Rule	Description
CALC AS	Calculate Assessment

Bill Msg	Description
SCB	School Board Operating

### Rate Schedule – MFCTR-1

Freq: **Monthly**    Currency: **USD**

RQ Rule	Description
CALC AS	Calculate Assessment

Bill Msg	Description
SCB	School Board Operating

RQ Rule	Description
NFMON	Non-filing fee based on # months

### Rate Schedule – Income -1

Freq: **Monthly**    Currency: **USD**

RQ Rule	Description
NFMON	Non-filing fee based on # months

**Duplicate versus Merge.** The [Rate Schedule](#) page has [Duplication](#) capability. You would duplicate a rate schedule if you want to a) create a new rate schedule AND b) populate it with all the information from an existing rate schedule. You would use the rate schedule merge page if you want to build a rate schedule using pieces of one or more rate schedules.

Open **Main Menu, Rates, Rate Schedule Merge** to open this page.

#### Description of Page

Select the **Original Rate Schedule** that is the target for merging the rate schedule collection information.

Select the **Merge From Rate Schedule** that is your template rate schedule to copy the collections from.

**Note.** You may only copy information from one Merge From rate schedule at a time. If you wish to copy information from more than one rate schedule, select the first Merge From rate schedule, copy the desired records, Save, then select the next Merge From rate schedule.

The left portion of the page will display any existing records in the collections for the original rate schedule. The right portion of the page will display the existing records in the collections for the Merge From rate schedule.

You may use the **Copy All** button to copy all the records in all the collections from the Merge From rate to the Original rate. If you do not choose to copy all, you may copy records individually as described below.

The left portion of the **RQ Rule** collection initially displays existing RQ rule records linked to the original rate schedule. In the **Merge Type**, you will see the word **Original**, for any of these records. The **Sequence** number and **RQ Rule** description are displayed. In the right portion of the collection, the existing records in the merge from rate are displayed initially.

The left portion of the **Bill Messages** collection initially displays existing bill messages linked to the original rate schedule. In the **Merge Type**, you will see the word **Original**, for any of these records. The description of each **Bill Message** is displayed. In the right portion of the collection, the existing records in the merge from rate are displayed initially.

The topics, which follow, describe how to perform common maintenance tasks:

## Contents

- [Removing A Row](#)
- [Adding A New Row](#)
- [Removing An Uncommitted Row](#)
- [Moving Rows Up and Down](#)

## Removing A Row

If you wish to remove a record linked to the Original rate schedule, press the “-“ button to the left of the record.

## Adding A New Row

You may move any of the records from the Merge From rate to the original rate by selecting the left arrow adjacent to the desired row. Once a record is moved it will disappear from the Merge From information and appear in the Original information with the word **Merge** in the Merge Type column.

## Removing An Uncommitted Row

If you have copied a row across by mistake, you may remove it by clicking on the right arrow adjacent to the appropriate record.

## Moving Rows Up and Down

The **RQ Rule** grids contain sequence numbers that indicate the order in which the rules should be executed. You may modify the execution order using the up and down arrows.

Refer to [Editable Grid](#) in the system wide standards documentation for more information about adding records to a collection by selecting from a list and repositioning rows within a grid.

## Setting Up Rate Factors

This section describes rate factors and how they are used to levy the many charges referenced in a rate.

**Warning!** Most rate factors exist to support calculations performed by rate components. This means that you must design your rate components before you can design your rate factors. We strongly recommend that you design “on paper” how every rate’s rate components looks before you attempt to set up rate factors.

### Contents

- [An Overview Of Rate factors](#)
- [The Structure Of A Rate Factor](#)
- [An Illustration Of A Rate Factor And Its Characteristics](#)
- [Deriving / Passing In Characteristic Values](#)
- [Defining Rate Factors](#)
- [Defining Rate Factor Values](#)

## An Overview Of Rate factors

The primary reason why rate components exist is to define how to calculate values to determine tax obligations. When you create rate components, the following factors indicate how these values should be calculated or determined:

- Specify the value directly in the rate component.
- Tell the rate component to use the value defined in a *rate factor*.
- Tell the rate component to use the value calculated by a “*for calculation purposes only*” rate component.
- Tell the rate component to call an algorithm to calculate the value.

The remainder of this section describes the rate factor approach. The other methods are discussed in [Designing Rate Components](#).

Rate factors are frequently used to specify a tax rate. For example, you could specify the current tax rate (say 6%) directly in every rate's rate components OR you could indicate every rate should levy the current state tax rate and have the system look up the applicable rate when it calculates bills. The latter approach involves using a rate factor to indicate that the amount to charge should be retrieved from someplace other than the rate component when a bill is calculated.

You can use rate factors to define any of the following types of charges:

- A flat amount. This would typically be used to define fixed charges such as fees.
- An amount for a given unit of measure.
- A percentage. This would typically be used for tax rates and exemption percents.

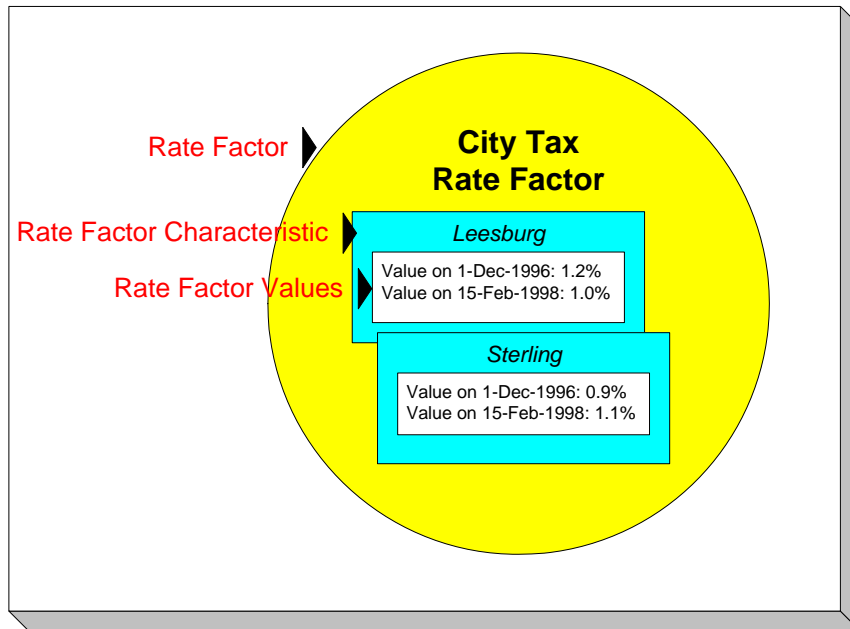
As a rule of thumb, you would use a rate factor (rather than specify the value in the rate component) when any of the following situations exist:

- The same charge exists in many rates. For example, if there are 3 rates that all contain the same fixed charge, it would make sense to use a rate factor to levy this charge rather than specify the same value on 3 rate components.
- The amount being charged is dependent on some physical characteristic of the taxpayer's location.
- The amount being charged is dictated by some external organization and therefore can change independently from the rate. For example, city tax percentages are set by city taxing authorities and can be changed at any time (and you don't want to change the rate when these charges change).
- The amount being charged varies depending on WHERE the taxpayer lives.
- A charge is only levied for a subset of taxpayers for defined periods of time.
- The charge differs per taxpayer.

**Warning!** As you can deduce, rate factors are extremely flexible. While the flexibility makes for a very powerful rating engine, it also presents you a dizzying array of options. We have discovered that to take advantage of the flexibility, you need to gain an overall understanding of the taxation provisions in ALL of your rates. Only then can you make logical and pragmatic decisions in respect of the number and type of rate factors.

## The Structure Of A Rate Factor

All rate factors share a common structure as illustrated below:

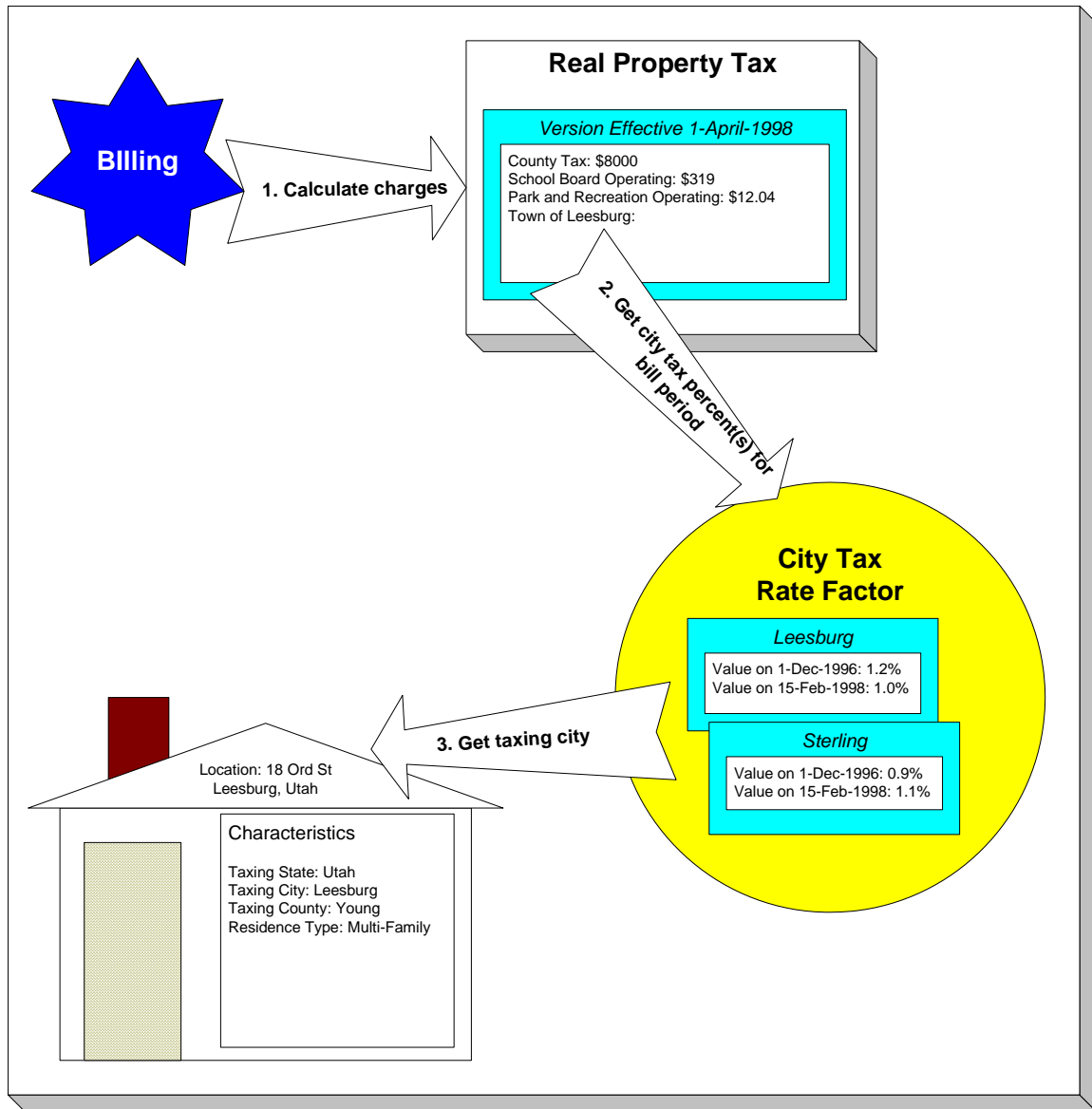


The above example shows a standard rate factor made up of the following tables:

- A **rate factor** contains descriptive information and attributes that control how the rate factor may be used in the system.
- All rate factors have the capability of having different values depending on some characteristic of the location. For example, a rate factor used to levy a city tax would have a different tax rate depending on the city in which the taxpayer resides. A **rate factor characteristic** must exist for every city with a tax (cities without a tax will not be linked to the city tax rate factor). At billing time, if the system cannot find a rate factor characteristic that corresponds with the taxpayer's city of residence, the charge will be skipped.
- And finally, each rate factor characteristic may have many **rate factor values** over time. To continue with our city tax example, each city with a tax can have different tax rates over time. There are different ways in which a rate factor's values may be defined:
  - For rate factors whose values are applicable to many taxpayers where only one value is effective on a given date, e.g., a city tax, the values are defined using rate factor values. Refer to [Defining Rate Factor Values](#) for more information.
  - For rate factors whose values are applicable to many taxpayers where more than one value is effective on a given date.
  - For rate factors whose value differs per taxpayer, the value is defined in the taxpayer's obligation. Refer to [Obligation – Contract Value](#) for more information about rate factors in contract terms.

## An Illustration Of A Rate Factor And Its Characteristics

The following picture illustrates how a rate factor and its characteristics are used to retrieve the relevant city tax at billing time:



The following points describe the above:

- At billing time, the billing engine sends a request to a rate to calculate the appropriate city tax based on the known real property tax calculated for the county.

- The rate calculates charges without rate factors until it encounters the rate component used to levy city tax. This rate component references a rate factor. This means the rate must get the tax rate(s) in effect during the billing period from this rate factor.
- The city tax rate factor contains an attribute defining that the location's taxing city characteristic controls the rate factor value. The rate factor therefore extracts the taxing city from the location.

**Deriving characteristic values.** Rather than have the system extract the characteristic value from an entity, you can setup the system to derive the characteristic value when the rate is calculated. For example, if all of your taxpayers are located in a single city, you may not want to maintain the taxing city on every location. To do this, you could setup the rate to "hard code" a taxing city of say Sterling. This is an advanced topic, but it may prove useful for your implementation. Refer to [Deriving / Passing In Characteristic Values](#) for the details.

- The location returns the taxing city and the rate factor extracts the tax rate(s) in effect during the bill period and returns them to the rate.
- The rate applies the tax percents and returns the charges to billing.

**Some rate factors don't need a characteristic.** There are rate factors whose value does not differ based on a characteristic. For example, a rate factor that does not need a characteristic is a fee assessed for returned checks. These are typically flat fees that do not vary when applied. However, because of the relational design of the system, every rate factor value must reference both a rate factor and a characteristic value. Therefore, if you have rate factors whose value is not related to a characteristic, you must specify a characteristic type on the rate factor with a characteristic value that is exactly the same as the characteristic type.

For more information about setting up characteristics, see [Setting Up Characteristic Types & Their Values](#).

## Deriving / Passing In Characteristic Values

**Warning!** This is an advanced, technical concept.



In [An Illustration Of A Rate Factor And Its Characteristics](#), we described how the system retrieves a characteristic value from a location when a rate factor is encountered that uses a location characteristic. This is a very common technique and is easy to understand. However, there are situations when an entity in Oracle Enterprise Taxation Management doesn't contain the appropriate characteristic value and therefore other techniques must be used. These techniques may not apply to your implementation, but they're worth understanding to help you form an intuitive understanding of rate application:

- Assume you have a charge that is based on the account's account type and the obligation's revenue class. This type of charge's characteristic value is not extractable from a single entity. Rather, the system must construct the characteristic value real-time by concatenating information from the account and obligation being billed. In this situation, you can use an RQ rule to derive this characteristic type and value. This RQ rule would create a characteristic type and value real time by concatenating the account's account type and the obligation's revenue class. This characteristic type and value only exists in memory while rate application executes. This is an example of the system deriving a characteristic value as opposed to it being passed in or retrieved from another object in the system.

When you setup a rate factor to use a characteristic whose value is derived or passed in, you need to define its characteristic source as **Characteristics Collection**. This source tells the system to extract the characteristic value from memory rather than extracting it from an object within the system.

**Note.** The various plug-in spots on a rate component have access to the **Characteristics Collection**. You might find this useful if you have a rate component that needs to calculate characteristic values for subsequent rate components.

## Defining Rate Factors

You create a rate factor for every type of charge whose value is not specified directly in a rate component.

You use two windows to set up a rate factor. On the first, you define general information describing how the rate factor is used in the system. On the second, you define every characteristic value for which the rate factor has a value.

### Contents

- [Defining General Rate Factor Information](#)
- [Setting Up Rate Factor Characteristics](#)

## Defining General Rate Factor Information

You begin to define a rate factor by selecting **Main Menu, Rates, Rate Factor**.

### Description of Page

Enter a unique **Rate Factor** code and a **Description** for the rate factor.

**Rate Factor Type** is set to **Regular**, to indicate that this rate factor's values are specified on [Rate Factor Value – Main](#) or on [Obligation – Contract Values](#).

Enter a **Currency Code** to define the currency in which the rate factor's charges are denominated. This field will be gray if the rate factor is referenced on a rate component.

**Note.** Rate factors referenced on a rate component must be denominated in the same currency as is the rate.

Select the **Value Type** to define the kind of value maintained by this rate factor. The Value Type options are:

<i>Charge</i>	The value maintained in this rate factor is a fixed amount (e.g., a flat charge).
<i>Percentage</i>	The value maintained in this rate factor is a percentage.
<i>Unit Rate</i>	The value maintained in the rate factor is a charge per rate quantity.

This field will be gray if the rate factor is already referenced on a rate component or if the rate factor already has values.

Turn on **Error If No Value** if the system should create a billing error when a rate factor value doesn't exist for the characteristic value linked to the taxpayer. Confused? To clarify this, you need to be aware that there are different types of rate factors:

- Those that should have a value for every taxpayer whose rate references the rate factor. A rate factor used to levy the relevant state tax is a good example (assuming every state has a tax). For this type of rate factor, if the system cannot find a rate factor value for the taxing state, it must generate a billing error because data is missing from the system. You would turn on **Error if No Value** for this type of rate factor.
- Those that only have a value when the taxpayer has a characteristic value that levies a charge. A rate factor used to levy the relevant city tax is a good example (because not all cities levy taxes). For this type of rate factor, if the system cannot find a rate factor value for the city in which the taxpayer lives, the rate component will be skipped and no billing error will be generated. You would turn off **Error if No Value** for this type of rate factor.

Rate factors used to levy taxes may or may not be eligible for tax exemptions; it's up to the respective taxing authority. Taxpayers who are eligible for the tax exemption will have their tax obligation reduced (or nullified) at billing time. For rate factors that are eligible for tax exemptions, turn on **Tax Exemption**. If an obligation is fully or partially exempt for the taxes defined by a given rate factor, the rate factor and the exemption information for the taxpayer must be indicated on the obligation page. This field will be gray if the rate factor is referenced as being tax exempt in an obligation.

**Tax Exemptions limited to Apply To Rate Components.** Although rate factors marked as eligible for tax exemptions may be referenced on several different types of rate components, only [Apply To](#) rate components include the logic to check for tax exemption eligibility.

For more information about how to make a taxpayer tax exempt, refer to [Obligation – Tax Exemptions](#).

Some types of rate factors do not have the same value for all taxpayers. Rather, their value may be unique for every taxpayer or it may be unique for some taxpayers and common for others. For example, rates may be defined for groups of taxpayers based on special incentive programs such as low-income groups or charitable organizations. Those taxpayers that do not qualify for these programs will use a general rate factor. For rate factor types where a taxpayer may have their own unique price, turn on **Value In Contract Term**. For any obligation that has a unique value for a given rate factor, the rate factor and the obligation's value must be indicated on the [obligation](#) page. When applying the rate for a rate factor of this type, the system will first check for the existence of a contract value for the obligation and if one is not found, it will use the value on the rate factor. This field will be gray if a contract term referencing this rate factor exists. Time of Use rate factors may follow this same logic. Taxpayers may override their time of use values by using a contract term.

**Note.** The **Error if No Value** switch also affects what happens if there is no contract term for the rate factor. If the error switch is on, a billing error will be generated. If the error switch is off, no billing error will be generated and the rate component will be skipped.

For more information about how to specify a rate factor value in a taxpayer's contract, refer to [Obligation – Rate Info](#).

Contract riders are used when the charge associated with the rate factor is only applicable to some taxpayers for some period of time. Turn on **Contract Rider Applicability** if the rate factor's applicability is contingent on the existence of a contract rider. This field will be gray if a contract rider referencing this rate factor exists on an obligation.

For more information about how to establish a contract rider for a taxpayer, refer to [Obligation – Contract Riders](#).

The proration information tells the system how to handle changes to rate factor values and their optional contract riders that may occur during a taxpayer's bill period. Turn on **Allow Proration** if changes to the rate factor's value and/or contract riders should be prorated. If changes should not be prorated, indicate the **Rate Selection Date** to use when the system retrieves rate factor values or determines if contract riders are applicable; available options are:

*Accounting Date*

The rate factor value effective on the bill segment's accounting date will be used.

**Bill End Date**

The rate factor value effective on the end date will be used.

**Bill Start Date**

The rate factor value effective on the bill segment's start date will be used.

**Rate Factor Tree**

The right half of this page is dedicated to a tree that shows the rate factors characteristics and their respective values. You can use this tree to both view high-level information about these objects and to transfer to the respective page in which an object is maintained by using the available Go To buttons.

**Where Used**

Many types of Rate Components may use a Rate Factor to define the amount being charged. Refer to [Rate Component – Main Information](#) for more information.

An obligation references Rate Factors as follows:

- When an obligation's **Apply To** rate component references tax exemptible rate factors AND the taxpayer is tax exempt, the obligation will define the rate factors from which the taxpayer is exempt. Refer to [Obligation – Tax Exemptions](#) for more information.
- When an obligation's rate references rate factors whose charges are taxpayer-specific (i.e., specified on a contract term), the obligation will contain charges for these rate factors. Refer to [Obligation – Contract Values](#) for more information.
- When an obligation's rate references rate factors that are contingent on contract riders, the obligation will define the rider period (if any) for these rate factors. Refer to [Obligation – Contract Riders](#) for more information.

**Setting Up Rate Factor Characteristics**

After defining general information about a rate factor, open **Main Menu, Rates, Rate Factor** and navigate to the **Rate Factor Characteristic** tab to define characteristic values for which a charge is applicable. For example, a rate factor used to levy the applicable city tax would require the definition of every city that levies a tax.

**Note.** If all values for a rate factor are specified in contract terms, you do not need to set up Rate Factor Characteristics.

**Description of Page**

Enter a **Characteristic Type** to indicate the type of characteristic that controls how the rate factor's value is defined. This field will be gray if Rate Factor Characteristics are specified in the following grid.

The **Characteristic Source** defines where the system retrieves the characteristic value at billing time. The Characteristic Source options are:

<i>Account</i>	The characteristic type's value will be retrieved from the obligation's account.
<i>Asset</i>	This option is not supported in this release.
<i>Characteristic Collection</i>	Refer to <a href="#">Deriving / Passing In Characteristic Values</a> for an explanation of this value.
<i>Main Person</i>	The characteristic type's value will be retrieved from the main person linked to the obligation's account.
<i>N/A</i>	The characteristic type's value is the same for all customers using the rate. If this option is used, you must still choose a Characteristic Type. This Characteristic Type should have a single Characteristic Value. The ID of the characteristic value must be identical to the ID of the characteristic type.
<i>Obligation</i>	The characteristic type's value will be retrieved from the obligation.
<i>Location</i>	The characteristic type's value will be retrieved from the obligation's characteristic location.

For more information about characteristics, see [Setting Up Characteristic Types & Their Values](#) and [An Illustration Of A Rate Factor And Its Characteristics](#).

Use the add and remove buttons to define the **Characteristic Values** that are relevant for the rate factor's Characteristic Type. For example, if the rate factor is used to levy a city tax, you would define the city's that have a city tax in the above collection.

Use the **External ID** if the values for this characteristic are interfaced to the system from an external source and you want to record an identifier for the source.

**Note.** If the rate factor's value is the same for all taxpayers, you still must link a "dummy" characteristic value to the rate factor. In other words, you'll have to create a characteristic type called "n/a" and define for it a characteristic value of "n/a". Then, on the rate factor, you reference the "n/a" characteristic type and on the above window, you indicate a characteristic value of "n/a".

## Defining Rate Factor Values

After the rate factor and its characteristic values are defined, you define the charge / percent and GL distribution associated with each rate factor characteristic. The topics in this section describe the pages used to set up this information.

### Contents

- [Rate Factor Value - Main](#)
- [Rate Factor Value - GL Distribution](#)

## Rate Factor Value - Main

After the rate factor and its characteristic values are defined, select **Main Menu, Rates, Rate Factor Values** to define the charge / percent applicable to each rate factor characteristic.

### Description of Page

Enter the **Effective Date** and respective **Value** for the rate factor and characteristic value. The first date should be the earliest date on which the rate factor's charges are effective. Subsequent effective dates and values are required whenever the value changes.

**Default note.** A **Value** of 0 will be defaulted.

## Rate Factor Value - GL Distribution

Select **Main Menu, Rates, Rate Factor Value** and navigate to the **GL Distribution** tab to define the GL account affected by the rate factor characteristic.

**Important!** If you do not specify this information for a rate factor characteristic, the distribution code defined on the rate component that references the rate factor will be used.

### Description of Page

If the rate factor characteristic's charges are booked to a different GL account depending on the characteristic value, specify the appropriate **Distribution Code** for each **Revenue Class**.

**Optional information.** Specifying distribution codes on a rate factor is optional. You would only do this if the GL distribution code differs based on the characteristic value (e.g., the city tax payable GL account differs depending on the city). If you do not enter the above information, the distribution code specified in the rate component that references the rate factor will be used.

## Defining Rate Versions

After defining general information about a rate on [Rate Schedule Maintenance](#), you must link to it a rate version. The rate version defines the effective date of the calculation rules defined in its rate components. A rate schedule will have multiple rate versions if its calculation rules (i.e., its rate components) change over time.

**A switch on the rate schedules controls what happens if multiple rate versions are effective during a bill period.** When the system creates a bill segment for an obligation, it checks if multiple rate versions are in effect during the bill period. If so, it uses the [rate schedule's Allow RVs Proration](#) switch and [RV Selection Date](#) flag to determine if it should prorate the various rate versions or if it should pick one of the rate versions effective during the bill period.

After a rate version exists, you add rate components to it using [Rate Component Maintenance](#) and/or [Rate Version Merge](#). After you have added all necessary rate components, don't forget to return to [Rate Version - Main](#) and change the state of the rate version to **Finished** (otherwise, it cannot be used by billing).

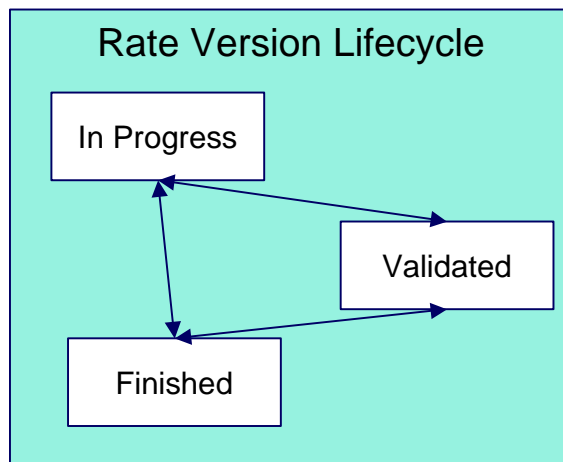
The topics in this section describe how to setup a rate version.

### Contents

- [Lifecycle of a Rate Version](#)
- [Rate Version - Main](#)
- [Rate Version - Bill Print Info](#)

## Lifecycle of a Rate Version

The following diagram shows the possible lifecycle of a rate version (you'll notice that you can change a rate version from any state to any state).



A rate version is in the **In Progress** state while you build its rate components. While a rate version is in this state, the system does not perform cross validation between rate components as they are added / changed / deleted using [Rate Component Maintenance](#) and/or [Rate Version Merge](#). However, the system will perform inter-field validation applicable to each type of rate component. Rate versions in this state may not be used by [Rate Check](#), nor will they be used up by the billing process.

You transition a rate version to the **Validated** state when you're ready to check it using [Rate Check](#), but you don't want billing to use it to calculate charges. When a rate version is transitioned to this state, the system performs cross validation between its rate components. If any validation errors are found, an error message is displayed and the status remains **In Progress**. If all validation checks are passed, the rate version status will become **Validated**. We'd like to stress that rate versions in this state may be used by [Rate Check](#), but will not be used by billing.

You transition a rate version to the **Finished** state when it can be used by billing to calculate charges. All of the cross validation between rate components will be performed when transitioning to this state.

**Finished and Validated states also affect rate component maintenance.** If a rate version is **Finished** or **Validated**, the system performs internal consistency checks every time a rate component is added, changed or deleted using [Rate Component Maintenance](#) and/or [Rate Version Merge](#). If you don't want these checks to be performed while you're working on a rate version's rate components, change its state to **In Progress**.

## Rate Version - Main

Use **Main Menu**, **Rates**, **Rate Version** to maintain a rate version.

### Description of Page

**Rate Version** contains basic information about the rate version. This information only appears after the rate version has been added to the database.

**Rate Schedule** defines the rate to which the rate version is linked. **Effective Date** defines the date on which the rate version's rate components become effective. These fields are the unique identifier of the rate version. Both fields become protected after the rate version is added to the database.

**Multiple rate versions may be prorated.** When the system creates a bill segment for an obligation, it checks if multiple rate versions are in effect during the bill period. If so, it uses the [rate schedule's Allow RVs Proration](#) switch and **RV Selection Date** flag to determine if it should prorate the various rate versions or if it should pick one of the rate versions effective during the bill period.

The **Rate Version Status** indicates the current state of the rate version. Refer to [Lifecycle of a Rate Version](#) for more information.



**Warning!** Billing ignores rate versions that are not in the **Finished** status. This means that if you create a new rate version and forget to finish it, billing will use the prior rate version when it calculates the taxpayers' bills.

Specify the description of the rate version in **Description on Bill**. This description is saved on bill calculation details created using this rate version. The following table illustrates the substitution variable(s) that may be used to substitute variables into the description during billing:

Substitution Character	What It Does
%D	Replaces the substitution character with message 10101 from message category 4. This message is maintained in the standard message table.  We recommend this message be set to "(prorated for %1 of %2 days)" where %1 will be substituted with the number of days of prorated consumption charged on the bill line and %2 will be the "normal" days associated with the rate's charges. This message will only be substituted when the rate version is prorated.

The following is an example of the content of common **Description on Bills** along with how the line would look on the bill:

Description on Bill	How It Looks On A Bill
Interest %D	Interest (prorated for 10 of 30 days)

The [tree](#) that follows contains a summary of the rate version's rate components. You can use this tree to both view high-level information about these objects and to transfer to the respective page on which an object is maintained.

**Note.** The duplicate action in the action button bar enables you to duplicate a rate version and its rate components. Refer to [Duplicate Button](#) in the system wide standards document for more information.

## Rate Version - Bill Print Info

This page allows you to maintain the bill print information for all of the rate version's rate components. Note, you can also maintain this information for individual rate components on [Rate Component – Main](#).

### Description of Page

The grid contains an entry for each rate component linked to the rate version. The following information is displayed in the grid:

**Sequence** uniquely identifies the rate component. It also controls the order in which the rate component is processed when the rate version is processed by billing.

**Description** is the rate component's description.

**Description on Bill** is the verbiage that appears on the bill line generated for this rate component. You can dynamically substitute variables into this description (e.g., charge and quantity) when billing calculates the rate component's charges. Refer to [How To Use Description on Bill](#) for a discussion of the various substitution variables.

Turn on the **Print** switch if you want the bill line to appear on the taxpayer's bill.

**Some bill lines don't print.** You may wonder why we give you the ability to not print a bill line. The reason is because ALL bill lines are shown on Bill Segment – Calculation Details. However, only those lines marked as Print are presented to the taxpayer. This way, if you want to suppress some calculation details, you can, but still show the details to your own employees.

The **Print If Zero** switch is only enabled if the **Print** switch is turned on. It allows you to indicate if a bill line should print when the bill line's value is zero:

- Turn this switch on if the bill line should print when the calculated value is zero.
- Turn this switch off if the bill line should not print when the calculated value is zero.

## How To Use Description on Bill

Enter the verbiage to appear on the bill segment in the **Description On Bill**. Turn on **Print** if the verbiage should appear on the printed bill.

**Suppressing calculation details on the printed bill.** Only rate components used to calculate intermediate values are typically suppressed on a bill. For these types of rate components, we recommend using **Description on Bill** to show users how the line was calculated, but turn off the **Print** switch. This way, users will see the calculation details on the [Bill Segment – Calc Lines](#) window, but the taxpayer won't see it on the bill.

Many lines that appear on a bill do not contain static verbiage. Rather, the printed lines contain information from the calculation process. For example, the bill line for state tax would typically contain the tax percent.

You control exactly what is substituted in the bill line by entering one or more of the following substitution variables in the **Description on Bill** field:

Substitution Character	What It Does
%A	Replaces the substitution character with the amount of the bill line. Please be aware of the following:

	<ul style="list-style-type: none"> <li>• If the rate component isn't designated as being "for calculation purposes only" (FCPO), all trailing zeroes up to the decimal position of the rate's currency will be removed. If the rate component is designated as being "for calculation purposes only" (FCPO), the number of decimal places as defined in the rate component's precision will be shown.</li> <li>• All leading zeroes will be removed unless the value is between 1 and -1 in which case a leading 0 will appear before the decimal place.</li> <li>• The Symbol on the Currency control table will prefix the value.</li> <li>• The position of the negative sign and currency symbol is dependent on the display profile of the user who generated the bill.</li> </ul>
%D	<p>Replaces the substitution character with message 10101 from message category 4. This message is maintained in the standard message table.</p> <p>We recommend this message be set to "(prorated for %1 of %2 days)" where %1 will be substituted with the number of days of prorated consumption charged on the bill line and %2 will be the "normal" days associated with the rate's charges. This message will only be substituted when the rate component is prorated. Note, neither %1 nor %2 are required.</p> <p>Either this character OR %F should be used when you desire to substitute the charge / percent into the bill line.</p> <p>If the charge is prorable, %D should be using in conjunction with %R to print an appropriate message when the charges are prorated.</p>
%F	<p>Replaces the substitution character with message 10102 from message category 4. This message is maintained in the standard message table.</p> <p>We recommend this message be set to "(prorated by %1)" where %1 will be substituted with the factor used to prorate the charge. This message will only be substituted when the rate component is prorated. Note, %1 is not required.</p> <p>If the charge is prorable, %F should be used in conjunction with %I to print an appropriate message when the charges are prorated.</p>
%I	<p>Replaces the substitution character with the value specified in the rate component / rate factor.</p> <p>If the value is a percent,</p> <ul style="list-style-type: none"> <li>• all leading and trailing zeroes will be suppressed,</li> <li>• and a "%" will be suffixed to the resultant value.</li> </ul> <p>If the value is not a percent,</p> <ul style="list-style-type: none"> <li>• all trailing zeroes up to the decimal position of the rate's currency will be removed,</li> <li>• and all leading zeroes will be removed unless the value is between 1 and -1 in which case a leading 0 will appear before the decimal place,</li> <li>• and note that the position of the negative sign and currency symbol is dependent on the display profile of the user who generated the bill.</li> </ul> <p>Either this character OR %R should be used when you desire to substitute the charge / percent into the bill line. Using %I will cause the value as specified in the rate component to be substituted regardless of whether the charges are prorated.</p>

	If the charge is prorable, %I should be used in conjunction with %F to print an appropriate message when the charges are prorated.
%Q	Replaces the substitution character with the rate quantity being billed. Leading zeroes are suppressed. The decimal places shown are specified in a field on the Unit Of Measure control table.
%R	Replaces the substitution character with the value specified in the rate component / rate factor. Either this character OR %I should be used when you desire to substitute the charge / percent into the bill line. Using %R causes the prorated charge / percent to be printed in the bill line (%I causes the charge / percent as specified in the rate to be printed during proration situations). If the charge is prorable, %R should be used in conjunction with %D to print an appropriate message when the charges are prorated.
%S	Replaces the substitution character with the summed amount. All trailing zeroes up to the decimal position of the rate's currency will be removed. All leading zeroes will be removed unless the value is between 1 and -1 in which case a leading 0 will appear before the decimal place. The Symbol on the Currency control table will prefix the value. If the value is negative, a "-" will prefix the value. This character should only be used with Minimum, Apply To, and Summary rate components.
%T	Replaces the substitution character with message 10103 from message category 4. This message is maintained in the standard message table. We recommend this message be set to "(%1% exempt)" where %1 will be substituted with the tax exemption percent as specified in the obligation. This message will only be substituted when the rate component's rate factor is eligible for tax exemptions AND the taxpayer is exempt from the tax. Note, %1 is not required.
Lines that don't print	If the bill line is not printed (as defined by the Print switch), message 10104 from message category 4 will be appended to the bill line. This message is maintained in the standard message table. We recommend this message be "NOT SEEN BY TAXPAYER!" This message appears in the bill line that appears in the billing windows but does not appear on the taxpayer's bill.

The following are examples of the content of common **Description on Bills** along with how the line would look on the taxpayer's bill:

Description on Bill On Rate Component	How It Looks On A Bill
County Tax at %R of %I: %A	County Tax at 1.0% of \$70,160: \$701.60
%O: %I %D	Homestead Exemption: \$150.24 (prorated for 100 of 183 days)
First %Q of Income %R	First \$45,000 of Income at 10%

State sales tax %R %T	State sales tax 6.25% (50% exempt)
Minimum charge of %I %F	Minimum charge of \$10.00 (prorated by .3333)
Subtotal of %S	Subtotal of \$1,123.23

The prorated references will only appear if the rate component's charges are prorated due to a short or long bill period.

## Designing Rate Components

Rate components are the fundamental building blocks of your rate calculation algorithms. All of the other rate and rate factor set up tasks are done in preparation of the creation of these records. This means that you must design your rate components before you can design rate factors, RQ rules, RQ identifiers, eligibility rules. We strongly recommend that you design “on paper” how every rate component looks on every rate schedule before you attempt to set up any of the rate control tables.

All of the calculation details and financial information that appear on a bill segment are derived using information on rate components. If you want to see the results of what rate components do, refer to [Bill Segment – Calc Lines](#).

**Warning!** There are innumerable ways to design rate components for a given rate. Some designs will result in easy long-term maintenance; others will result in maintenance headaches. In this section, we provide information to help you understand the ramifications of the various options. Before you set up your production rate components, we encourage you to gain intuitive understanding of these options by using the system to prototype the alternatives.

The topics in this section provide background information that will facilitate the construction of your rate components.

### Contents

[Rate Component Design Methodology](#)

[Rate Component Rounding](#)

[The Big Picture Of Rate Component Eligibility Rules](#)

## Rate Component Design Methodology

Designing rate components is an iterative process. Over time, you will develop intuitive skills that will allow you to avoid some iterations. However, when you're starting out, we recommend you follow these steps to design your rate components:

- Obtain copies of existing bills that use the rate in question. If the rate is new, then write up EXACTLY how the information should appear on the taxpayers' printed bills. If the rate has optional charges (e.g., city taxes that are added if applicable or senior citizen discounts), make sure your examples encompass every possible scenario.

- Next, look at the bill lines and ask yourself what are the variables that cause each line's charges to be calculated the way they are. Consider the following examples:
  - "County Tax at 1.0% of \$70,160: \$701.60". This charge is based on the tax rate and the base amount for the calculation. The variables involved are the tax rate, the base amount and the resulting tax amount.
  - "Homestead Exemption: \$150.24 (prorated for 100 of 183 days)". This line may appear on the bill if the taxpayer qualifies for this particular type of exemption. The variables involved are the exemption type, amount and proration (if applicable).
  - "Minimum monthly charge of \$100.00". This is a charge that only appears if the total of the prior lines is less than \$100.00. The variables involved in this line are a little complicated because they must be calculated at billing time by adding up several other lines and comparing them to the minimum charge amount. This means the variables are the total of prior lines and the minimum charge amount.
  - "1.25% County Surcharge". This charge is similar to the minimum charge in that it is calculated by adding up one or more prior lines and applying a percentage to the sum. This means the variables are the total of prior lines plus the tax percentage.
- After you've determined all of the potential bill lines you can start designing your rate components. Typically you create one rate component for every line that can appear on the taxpayer's bill. When you create a rate component, you will categorize it as one of the following types:

*Flat Charge* This type of rate component is used to create bill lines that levy fixed charges and fees that aren't based on the taxpayer's income, asset value or other assessable amounts. The monthly charge shown above would be levied using this type of rate component.

*Rate Quantity* This type of rate component is used to create bill lines that levy charges based on some type of input quantity.

*Apply To* This type of rate component is used to create bill lines that levy charges based on the amount calculated on other bill lines.

*Summary* This type of rate component is used to create a "subtotal" on the bill. It exists purely for aesthetic purposes.

*Minimum Charge* This type of rate component is used to create bill lines that levy charges only when the sum of previously calculated lines is less than the minimum charge amount. The minimum monthly charge shown above would be levied using this type of rate component.

Note that if the values being compared are negative values, the comparison is NOT done on the absolute values, but rather on the actual values. For example, imagine you have a minimum discount of \$ -2.00 and previous rate components have calculated a discount of \$ -1.00. You want a rate component to create a bill line for \$ -1.00 to apply a further discount. A minimum charge rate component will not work in this case because -1 is considered more than -2. For this business scenario you should use a maximum charge rate component.

<i>Maximum Charge</i>	<p>This type of rate component is used to create bill lines that levy charges only when the sum of previously calculated lines is more than the maximum charge amount.</p> <p>Note that if the values being compared are negative values, the comparison is NOT done on the absolute values, but rather on the actual values. For example, imagine you have a maximum discount of \$ -2.00 and previous rate components have calculated a discount of \$ -3.00. You want a rate component to create a bill line for \$1.00 to reduce the discount. A maximum charge rate component will not work in this case because -3 is considered less than -2. For this business scenario you should use a minimum charge rate component.</p>
<i>Exact Charge</i>	This type of rate component is used to force a bill to add up to a given amount (regardless of how much the bill would be based on earlier rate components).
<i>Calculation Algorithm</i>	This type of rate component enables you to produce bill calculation lines based on logic in an algorithm that you supply. Use this rate component when none of the other rate component functions will provide you with the logic you require.

- For each rate component, determine if the taxpayer must meet some form of eligibility criteria before the rate component is applied. For example, you may have a rate component that is calculated if a taxpayer is a senior citizen. This rate component may have eligibility criteria that requires the person to have a given characteristic value before the rate component is applied. Refer to [The Big Picture of Rate Component Eligibility](#) for more information.
- For each type of rate component (except summary), determine how you'll define its unit rate / percent / flat amount. You have four choices:
  - Specify the value directly in the rate component.
  - Tell the rate component to use the value defined in a *rate factor*.
  - Tell the rate component to use the value calculated by a “*for calculation purposes only*” rate component.
  - Tell the rate component to call an algorithm. The algorithm will return the value.

The following tables provide guidance in respect of which of the methods to use:

<i>Specify value in the rate component</i>	<p>You'd specify the value in the rate component when the value in the rate is specific to that rate and is applied to all taxpayers regardless. For example, if you see a provision in the rate to add a trash fee of \$45, you'd specify 45.00 directly in the rate.</p>
<i>Use a rate factor to define the value</i>	<p>The following paragraphs describe when you'd use a rate factor to define a unit rate / percent / flat amount.</p> <p>The same charge exists in many rates.</p> <p>The amount being charged varies depending on where the taxpayer lives. If you have a rate where the charge differs depending on where the taxpayer lives, you should use a rate factor to levy this type of charge.</p> <p>A charge (or discount) is only levied for a subset of taxpayers for defined periods of time.</p> <p>The charge differs per taxpayer.</p>

***Use a "for calculation purposes only" rate component***

This is complicated to explain and is not used very often, but it is very powerful and handles some very complicated algorithms. It works like this – you calculate the charge / percent / flat charge on another rate component (marked as "for calculation purposes only" so it won't affect the bill amount), and then reference it on the "real" rate component.

***Call an algorithm***

This is complicated to explain and is not used very often, but it is very powerful and handles some very complicated examples. It works like this – you call an algorithm and it calculates the charge / percent / charge.

We supply an example of one such algorithm in the base package. If your rates require additional algorithms, you will have to develop additional algorithms. Refer to [How to set up rate quantity rate components](#) under [Rate Component – Main Information](#) for more information about this type of algorithm.

- After you've categorized the bill lines into the various types and know how the price is defined, you determine how each line's charges are reflected in your general ledger. You will need to work with your accounting department as they will tell you the exact revenue, expense, and liability accounts that will be affected by the charges.
- Finally, read the provisions of your rate (the legal wording) and make sure you haven't left out something. This may involve having to discuss confusing provisions with your legal department.

At this point, you're ready to start entering your rate components.

## Rate Component Rounding

The topics in this section describe how to control how each rate component's final value is rounded.

### Contents

[Rounding Precision Is Defined On A Rate Component](#)  
[Rounding Method Is Defined On A Rate Component](#)  
[Rounding And FCPO Rate Components](#)  
[Interim Rounding For Apply To Rate Components](#)  
[Rounding At The End](#)

### Rounding Precision Is Defined On A Rate Component

A rate schedule references a currency code. A rate's currency code defines its maximum number of decimal places. These decimal places, in turn, control the smallest unit to which most rate components can be rounded (i.e., the rate component's precision). For example,



- If a rate's currency has 2 decimal places, the smallest rounding precision that most of its rate components can have is 0.01. In other words, the smallest unit to which most dollar-based rate components can be rounded is to the cent. Note, whether you round up / down / to the nearest is discussed below.
- If a rate's currency has no decimal places, the smallest rounding precision that most of its rate components can have is 1.00.

The system supports the notion of a rate component having a greater rounding precision than its currency. For example, a rate's currency may support 2 decimal places, but you can setup rate components to have a rounding precision of 0.05.

The reason that we underlined the word most in the previous paragraphs is that these rules do not apply to "for calculation purposes only" (FCPO) rate components. FCPO rate components don't contribute real amounts to a bill (they exist to calculate intermediate results that are used by later rate components) and therefore they can support a rounding precision of up to 0.00001 (5 decimal places).

### Rounding Method Is Defined On A Rate Component

When you setup a rate component you must define the rounding method. You are given the following choices:

- ***Always round up.*** This rounding method rounds a rate component up to the nearest value (as controlled by the rate component's precision). For example, if a rate component's precision is 0.01 and a value of 0.011 is calculated, the rate component's final value would be rounded up to 0.02.
- ***Always round down.*** This rounding method rounds a rate component down to the nearest value (as controlled by the rate component's precision). For example, if a rate component's precision is 0.01 and a value of 0.019 is calculated, the rate component's final value would be rounded down to 0.01.
- ***Round to the nearest value.*** This rounding method rounds a rate component to the nearest value (as controlled by the rate component's precision). For example, if a rate component's precision is 0.01 and a value of 0.019 is calculated, the rate component's final value would be rounded up to 0.02. Whereas, if a value of 0.012 is calculated, the rate component's final value would be rounded down to 0.01.

### Rounding And FCPO Rate Components

As described above, for calculation purpose only (FCPO) rate components don't contribute real amounts to a bill. FCPO rate components exist to calculate intermediate results that are used by later rate components. We therefore allow FCPO rate components to have a rounding precision greater than the rate's currency code to a maximum of 0.00001 (5 decimal places).

Please be aware of the following:

- As described under [How To Use Description on Bill](#), you can indicate that the final amount of a bill line should be substituted into the bill line's description by using the %A substitution variable in the rate component's **Description On Bill**. If you do this for an FCPO rate component, the system will show the number of decimal places as dictated by the FCPO's precision.
- While rate application calculates bill lines, the system maintains each rate component's value in memory. For FCPO rate components, this means that precisions greater than .01 are available during rate application. For example, you could indicate an FCPO rate component has a precision of .0001 and the resultant value will be maintained in memory and is available to other rate components while rate application executes.

When rate application completes, the FCPO's value will be either saved in the bill line's amount or discarded as per the value of the rate component's **FCPO retention rule**. If you indicate that the FCPO amount should be retained on the bill line and its precision is greater than 2 decimal places, the FCPO amount will be rounded to two decimal places before it's saved on the database. Keep in mind that this could result in an inconsistency if you used the %A substitution variable in the rate component's **Description On Bill** as the value substituted into the bill line will have a greater precision.

### Interim Rounding For Apply To Rate Components

The calculations for Apply To rate components involve several steps with interim values stored along the way. For every interim value that is stored, rounding rules are applied. The following steps describe the interim rounding rules:

- If the value type is not **Unit Rate**, the amounts for all the "cross-reference" rate components are added together and stored in the database as the Base Amount. The base amount is stored with a precision of .01 and it is rounded to the nearest value based on the currency's precision.
- The rate component calculates its charge and keeps the result in a temporary field with a precision of .00001.
- The remaining steps depend on whether or not a tax exemption is applicable
  - If NO tax exemptions are applicable, the temporary field is rounded based on the rate component's rounding rules to produce the final amount.
  - If there are tax exemptions, the temporary field is rounded to the nearest .01 decimal precision prior to applying the tax exemptions. When the tax exemption is applied, the result is rounded according to the currency precision and stored as the Exempt Amount. The system calculates the final amount by subtracting the exempt amount from the rounded temporary amount and then applying the rate component's rounding rules.

## Rounding At The End

The following is an example of a simple bill segment with three bill lines.

Bill Segment - Obligation1		
Line Description	Amount	GL Impact
Usage charge	\$501.00	Revenue \$501.00
Tax	\$5.01	Tax payable \$5.01
Total	\$506.05	Rounding \$0.04

The first two bill lines are calculated using simple rate components that have a precision of .01. The last bill line is special, it rounds the bill up to the next highest 0.05 (this is an example from a country that doesn't have cents, their smallest coin is 0.05). The following would be necessary to calculate the last bill line:

- The rate component would be an exact charge rate component
- Its value (i.e., the exact charge) would be calculated using a value calculation algorithm that sums the first two rate components and rounds them up to the nearest .05. Refer to [RCVL-RNDXRF](#) for a value calculation algorithm that can perform this function.

An alternative approach is to do the following:

- Use a summary rate component to calculate the exact charge. This rate component would require the following characteristics:
  - FCPO
  - Non printing
  - Rounding type would be round up
  - Precision would be 0.05
- Reference the summary rate component as the exact charge rate component's value

## The Big Picture Of Rate Component Eligibility Rules

You can use eligibility rules to define the criteria when a rate component should be applied (or skipped). For example,

- For example, you may have a rate component that is calculated if a taxpayer is a senior citizen. This rate component may have eligibility criteria that require the person to have a given characteristic type and value (this assumes you use a person characteristic to capture the taxpayer's date of birth) before the rate component is applied.
- You may have a rate component that calculates a surcharge if the taxpayer uses 25% more than they used in the same period in prior year.

**Eligibility rules are optional.** If you put eligibility rules on a [rate component](#), the rate component will be skipped or applied as per the instructions in the rules. If you don't put eligibility rules on a rate component, the rate component will be executed (i.e., rate components are eligible by default).

The topics in this section describe eligibility rules.

### Contents

- [A Rate Component Is Eligible By Default](#)
- [Criteria Groups versus Eligibility Criteria](#)
- [Defining Logical Criteria](#)
- [Examples Of Rate Component Eligibility Rules](#)

## A Rate Component Is Eligible By Default

If you don't want to setup eligibility rules, you don't have to. If you don't specify eligibility rules on a rate component, it's eligible (i.e., it's going to be processed by rate application).

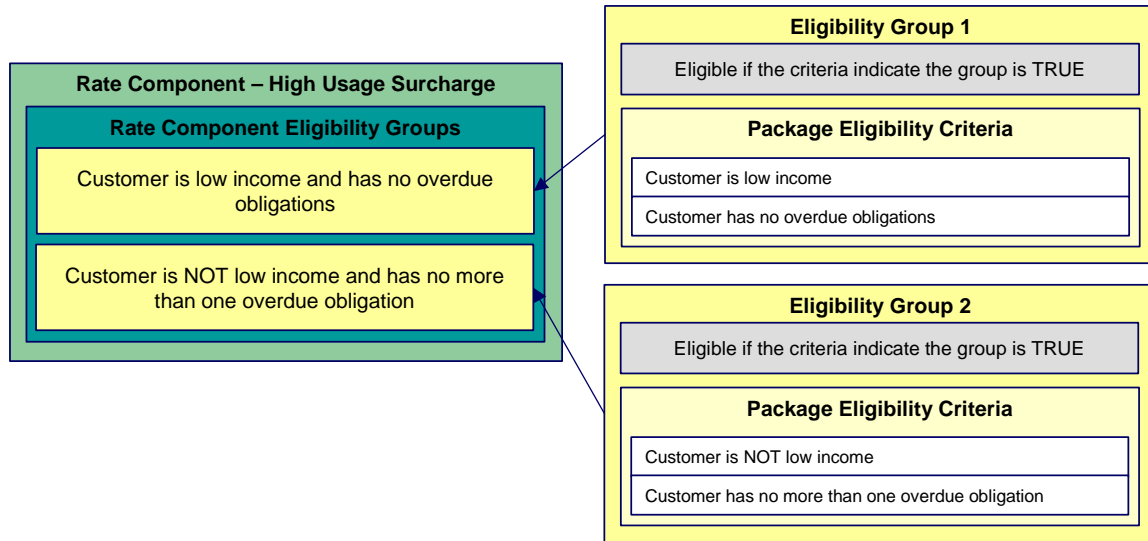
## Criteria Groups versus Eligibility Criteria

Before we provide examples of eligibility criteria, we need to explain two concepts: Criteria Groups and Eligibility Criteria. A rate component's criteria groups control whether rate application executes a rate component. At a high level, it works like this:

- A criteria group has one or more eligibility criteria. A group's criteria control whether the group is considered TRUE or FALSE.
- When you create a group, you define what should happen if the group is TRUE or FALSE. You have the following choices:
  - The rate component should be applied
  - The rate component should be skipped
  - The next group should be checked

We'll use the following example to help illustrate these points. Assume a rate component is only eligible if:

- The taxpayer is a low income taxpayer and has no overdue obligations.
- OR, the taxpayer isn't a low income taxpayer and has no more than one overdue obligation.



This rate component requires two eligibility groups because it has two distinct conditions:

- IF (Taxpayer is low income AND the taxpayer has no overdue obligations)
- IF (Taxpayer is NOT low income AND the taxpayer has no more than one overdue obligation)

If either condition is true, then the rate component is eligible.

You'd need to setup the following criteria groups in order to support this requirement:

Group No.	Group Description	If Group is TRUE	If Group is FALSE
1	Taxpayer is a low income taxpayer and has no overdue obligations	<i>Eligible</i>	<i>Check next group</i>
2	Taxpayer is NOT a low income taxpayer and has no more than one overdue obligation	<i>Eligible</i>	<i>Ineligible</i>

The following criteria will be required for each of the above groups:

Group 1: Taxpayer is a low income taxpayer and has no overdue obligations				
Seq	Logical Criteria	If Eligibility Criteria is TRUE	If Eligibility Criteria is FALSE	If Insufficient Data
10	Taxpayer is low income	<i>Check next condition</i>	<i>Group is false</i>	<i>Group is false</i>
20	Taxpayer has no overdue obligations	<i>Group is true</i>	<i>Group is false</i>	<i>Group is false</i>

Group 2: Taxpayer is NOT a low income taxpayer and has no more than one overdue obligation				
Seq	Logical Criteria	If Eligibility	If Eligibility	If Insufficient

		Criteria is TRUE	Criteria is FALSE	Data
10	Taxpayer is NOT low income	<i>Check next condition</i>	<i>Group is false</i>	<i>Group is false</i>
20	Taxpayer has no more than one overdue obligation	<i>Group is true</i>	<i>Group is false</i>	<i>Group is false</i>

The next section describes how you'd setup the specific logical criteria in each of the groups.

### Defining Logical Criteria

When you setup an eligibility criterion, you must define two things:

- The field to be compared
- The comparison method

You have the following choices in respect of identifying the ***field to be compared***:

- You can retrieve a characteristic value linked to any of the following:
  - The obligation being billed
  - The obligation's account
  - The main person linked to the obligation's account
  - The characteristic location linked to the obligation
  - The obligation's asset (not supported in this release)
  - In addition, you can also use a characteristic value that is derived while the rate is being calculated (characteristic values can be created by RQ rules and many other rate component algorithms)
- You can retrieve the value of a given rate quantity
- You can retrieve the final value of an earlier rate component
- You can execute an algorithm to retrieve a field value from someplace else in the system. This is a very powerful feature, but it's not terribly intuitive. We'll present a few examples later in this section to illustrate the power of this approach.

You have the following choices in respect of identifying the ***comparison method***:

- You can choose an operator (e.g., >, <, =, BETWEEN, IN, etc.) and a comparison value.

- You can execute an algorithm that performs the comparison (and returns TRUE, FALSE or INSUFFICIENT DATA). This is also a very powerful feature, but it's not terribly intuitive. We'll present a few examples later in this section to illustrate the power of this approach.

The [Examples Of Rate Component Eligibility Rules](#) provide examples to help you understand this design.

## Examples Of Rate Component Eligibility Rules

The topics in this section provide examples about how to setup rate component eligibility rules.

### Contents

[A Rate Component With A Time Span Comparison](#)

[A Rate Component With Tax Type Comparison](#)

### A Rate Component With A Time Span Comparison

A rate component that is only eligible for senior citizens has the following eligibility rule:

- Birth date equates to that of a senior citizen

This rule requires only one eligibility group on the rate component. It would look as follows:

Group No.	Group Description	If Group is TRUE	If Group is FALSE
1	Taxpayer Is A Senior Citizen	<i>Apply rate component</i>	<i>Skip rate component</i>

The following criterion is required for this group:

Group 1: Residential, Calif, Senior					
Seq	Field to Compare	Comparison Method	If TRUE	If FALSE	If Insufficient Data
10	Main person characteristic: Date of Birth	Algorithm: True if senior	<i>Group is true</i>	<i>Group is false</i>	<i>Group is false</i>

The criterion contains a time span comparison. Time span comparisons are used to compare a date to something. In our example, we have to determine the age of the taxpayer based on their birth date. If the resultant age is > 65, they are considered to be a senior citizen. To pull this off, you can take advantage of a comparison algorithm supplied with the base rate component as described below.

- Field to Compare. The person characteristic in which the taxpayer's birth date is held is selected.
- Comparison Method. We chose a comparison algorithm that returns a value of **TRUE** if the related field value (the taxpayer's date of birth) is greater than 65 years (refer to [RECC-TIMESPN](#) for an example of this type of algorithm).

You'll notice that if a value of **TRUE** is returned by the **True if senior** algorithm, the group is true (and we've setup the group to indicate a true group means the rate component is eligible).

**The time span algorithm can be used to compare days, weeks, months, etc.** Refer to [RECC-TIMESPN](#) for more information about this algorithm.

#### A Rate Component With Tax Type Comparison

Imagine a rate component that is only eligible if the current taxpayer has two tax types. This rate component would need the following eligibility rules:

- Taxpayer has real property taxes, and
- Taxpayer has personal property taxes.

These rules require only one eligibility group on the rate component. It would look as follows:

Group No.	Group Description	If Group is TRUE	If Group is FALSE
1	Has real property AND personal property taxes	<i>Apply rate component</i>	<i>Skip rate component</i>

The following criteria will be required for this group:

Group 1: Has real property AND personal property taxes					
Seq	Field to Compare	Comparison Method	If TRUE	If FALSE	If Insufficient Data
10	Algorithm: check if taxpayer has real property taxes	= TRUE	<i>Check next condition</i>	<i>Group is false</i>	<i>Group is false</i>
20	Algorithm: check if taxpayer has personal property taxes	= TRUE	<i>Group is true</i>	<i>Group is false</i>	<i>Group is false</i>

Both criteria are similar – they call an algorithm that performs a logical comparison. These algorithms are a bit counter intuitive (but understanding them will provide you with another way to implement complex eligibility criteria):

Both criterion works as follows:



- **Field to Compare.** We chose a “field to compare” algorithm that checks if the current account has obligations that belong to a given set of tax types. It returns a value of **TRUE** if the taxpayer has an active obligation that matches one of the tax types in the algorithm. In our example, the “check if taxpayer has real property taxes” algorithm returns a value of **TRUE** if the taxpayer has at least one active obligation whose obligation type references the real property tax type. The “check if taxpayer has property taxes” algorithm is almost identical, only the tax type differs. Refer to [RECF-SRVTY](#) and [RECF-AUTOPAY](#) for examples of this type of algorithm.
- **Comparison Method.** We simply compare the value returned by the algorithm to TRUE and indicate the appropriate response.

**Bottom line.** The “field to compare” algorithm isn’t actually returning a specific field’s value. Rather, it’s returning a value of **TRUE** or **FALS** (this is not a misspelling). This value is, in turn, compared by the “comparison method” and the group is set to true, false or check next accordingly.

## Defining Rate Components

We strongly recommend that you familiarize yourself with the information under [Designing Rate Components](#) before using this transaction.

Rate components control exactly how your bill calculation lines are calculated. Every rate component exist in respect of a given [rate version](#). This is because the rate version defines the date on which all of its rate components become effective.

**Copying, delete and moving rate components.** You can use the rate component maintenance transaction to duplicate, delete and move rate components. However, you may find these functions are easier to do using the [Rate Version Merge](#) transaction as this transaction can copy en masse and it also updates the relevant cross reference information if you move or delete a rate component.

The topics in this section describe how to maintain a rate component.

**Don’t forget to finish the rate version.** After you have added all necessary rate components to a rate version, don’t forget to return to [Rate Version - Main](#) and change the state of the rate version to **Finished** (otherwise, it cannot be used by billing).

### Contents

- [Rate Component - Main Information](#)
- [Rate Component - Cross Reference](#)
- [Rate Component - GL Distribution](#)

[Rate Component - Characteristics](#)  
[Rate Component - Eligibility](#)

## Rate Component - Main Information

Core information about a rate component is defined using **Main Menu, Rates, Rate Component**.

### Description of Page

**Rate Component** contains a concatenation of basic information about the rate component. This information only appears after the rate component has been added to the database. The adjacent up and down arrows cause the rate component immediately before or after this rate component to be displayed.

**Rate Version** defines the rate version to which the rate component is linked. **Sequence** defines the relative position of this rate component in respect of the other rate components. The position is important because it defines the order in which the system calculates the charges AND the order in which the calculated charges appear on bills. **Rate Version** and **Sequence** form the unique identifier of the rate component. Both fields become protected after the rate component is added to the database.

**Re-sequencing rate components.** You must use the [Rate Version Merge](#) transaction to reposition a rate component. If you use the up / down arrows on the [Rate Version Merge](#), the system will change the rate component's **Sequence** and change all rate components that reference the old **Sequence** accordingly.

**Leave gaps in the sequence numbers.** Make sure you leave space between sequence numbers so that you can add new rate components between existing ones in the future. If you run out of space between rate components, you can use the [Rate Version Merge](#) transaction to re-sequence a rate version's rate components.

**Eligibility criteria are highlighted.** If the rate component has [eligibility criteria](#), an indication of such appears.

Select the **Rate Component Type** that corresponds with the rate component. The screen will display the appropriate fields, which correspond with the categories of rate components described in [Designing Rate Components](#). This field will be gray when the rate component is referenced on other rate components.

**Warning!** The Rate Component Type affects what you can enter on other parts of the window. The remainder of this section is devoted to those fields that can be entered regardless of RC Type. The subtopics that follow describe those fields whose entry is contingent on the RC Type.

Use **Description** to describe what the rate component does.

**Rounding Type** and **Precision** control how the system rounds the rate component's calculated value. Refer to [Rate Component Rounding](#) for a complete description.

**Default note.** **Rounding Type** defaults to **Nearest** and **Precision** defaults to a value consistent with the decimal positions defined on the rate schedule's currency code.

Turn on **For Calculation Purposes Only** (FCPO) if this rate component exists purely to calculate the percent / flat rate / unit rate used by another rate component. When this switch is on, the system does not include the calculated amount in the bill total.

When this switch is on:

- You must define what is calculated in **Result Type**; permissible values are **Unit Rate**, **Charge**, **Step Multiplier** and **Percentage**.
- Use **Create a Bill Line** to define if a bill line should be created for the rate component (if you don't need to show the results to the taxpayer or a user, there is no need to create a bill line).
- If a bill line is being created, use the **FCPO Retention Rule** to define if the FCPO amount should be set to zero on the bill line or whether the bill line's amount should be set equal to the FCPO amount.

For each rate component, you will need to select the **Value Type** and **Value Source**. Valid values for the Value Type are **Charge Percentage** and **Unit Rate**. More information to help determine which value type to choose are described in the How To sections below. Based on the Value Source entered, the remainder of this row will change.

If the Value Source is **Rate Factor**, then a prompt to indicate the **Rate Factor** will appear.

If the Value Source is **Other Rate Component**, then a prompt to indicate the **RC Sequence** will appear.

If the Value Source is **Value**, then a prompt to indicate the **Value** directly on the rate component will appear.

If the Value Source is **Value Algorithm**, then a prompt to indicate the **Value Algorithm** will appear.

If you plan to use this method, you must set up this algorithm in the system. This can be done with the following options:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that calculates a rate component's value. Click [here](#) to see the algorithm types available for this plug-in spot.

For more information about which method to use, refer to [Designing Rate Components](#).

If the charges associated with the rate component are only levied during a specific season, you must turn on the **Seasonal** options.

When **Seasonal** is turned on, you use **Prorate Method** to define how the system should prorate a seasonal charge when a bill segment's start and end dates are not entirely within the seasonal period (defined in **Season**). The following options are available:

- **Accounting Date Dependent.** This option will not prorate the charge calculated by this rate component. Rather, it will only calculate the charge if the accounting date associated with the bill segment is within the seasonal period.
- **Bill End Date.** This option will not prorate the charge calculated by this rate component. Rather, it will only calculate the charge if the bill segment end date is within the seasonal period.
- **Bill Start Date.** This option will not prorate the charge calculated by this rate component. Rather, it will only calculate the charge if the bill segment start date is within the seasonal period.
- **Prorate.** This option will prorate the charge calculated by this rate component based on the number of days in the adjacent season that is included in the bill period. For example, if a bill period is from 15-April through 15-May and the seasonal period is from 1-May through 31-October, the system will assume 15 / 30 of the charge should be levied.
- **Prorate seasonal RQ.** This option is only pertinent if you have:
  - Seasonal rate quantity (RQ) charges, AND
  - You have a bill that crosses seasonal boundaries, AND
  - You have multiple rate versions effective across the bill period.

If this option is used, the system will prorate the charge calculated by a RQ rate component based on the number of days in the adjacent season that is included in the rate version's calculation period.

**Note.** The **Seasonal** period is defined in the two adjacent fields. The first field contains the day and month when the season starts; the second field contains the day and month when the season ends. The day and month should be entered in the format defined in your [display profile](#).

**Override Seasonal Proration.** If the seasonal functionality provided with the system does not work for your organization, you may override the logic using the **Override Seasonal Proration** plug-in spot on the [installation](#) record. For example, perhaps the seasonal period is determined dynamically based on the scheduled meter read date of the bill cycle. Or perhaps a seasonal rate component should not be prorated and the date used to determine whether the rate component is applicable is not in the list provided by the system.

**Description On Bill, Print, and Print If Zero** all control if the rate component contributes a line to the taxpayer's bill. These fields can also be modified on [Rate Version - Bill Print Info](#). You might find it easier to setup these fields on the rate version transaction as you can copy and paste descriptions between rate components. Refer to [Rate Version - Bill Print Info](#) for a description of these fields.

The other fields on this page are dependent on the type of rate component. See the "how to" subtopics below for more information.

## How To

### Contents

- [How To Set Up Flat Charge Rate Components](#)
- [How To Set Up Rate Quantity Rate Components](#)
- [How To Set Up Apply To Rate Components](#)
- [How To Set Up Maximum Charge Rate Components](#)
- [How To Set Up Minimum Charge Rate Components](#)
- [How To Set Up Exact Charge Rate Components](#)
- [How To Set Up Summary Rate Components](#)
- [How To Set Up Calculation Algorithm Rate Components](#)

### How To Set Up Flat Charge Rate Components

There are no special additional fields on this page available for setting up a Flat rate component. The following information will help you determine how to set up your Flat rate components.

The **Value Type** in this case would be **Charge**. This field will be gray when the rate component is referenced on another rate component.

Refer to [Rate Component - Main Information](#) for information about defining the Value Source.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **GL Distribution** window to define how to book moneys associated with this rate component in the general ledger.

### How To Set Up Rate Quantity Rate Components

When setting up an RQ rate component, additional fields become available for you to define.

The following information will help you to set up your RQ rate components.

Select a **Value Type** of **Unit Rate** if the type of charge is an amount per some unit/quantity. In the rare situation where the amount of the charge is flat for the first X units, use the Value Type of **Charge**. This field will be gray when the rate component is referenced on another rate component.

Refer to [Rate Component - Main Information](#) for information about defining the Value Source.

The amount that you specified using the rate factor / value field was a rate per some unit of some thing. This thing must be identified using the **Unit Of Measure**, **Time Of Use**, and **Rate Qty. Identifier** fields.

Turn on **Measures Peak Qty** if you do not want this rate component to prorate.

**Default note.** The value of **Measures Peak Qty** is defaulted from the **UOM** table (if a UOM is specified).

Turn on **GL Statistical Quantity** if GL journal lines generated for this rate component should also contain the rate quantity amount as a statistical quantity. You would use this option if you keep track of both dollar amounts and consumption units in your general ledger.

**Warning!** When a bill segment is created, the system stores the statistical quantity on the journal line associated with the rate component's distribution code. If you book multiple UOM's to the same distribution code, you should only turn on GL Statistical Qty on one of the rate components. Otherwise, you will commingle different UOM's on a journal line's statistical quantity.

If the charge is only applicable to a quantity within some tier, indicate that the rate component is **Stepped**. When this option is turned on, you must indicate the **Step Low Value** and **Step High Value** that the charge is applicable to. When multiple tiers exist, the high value of the first tier is the low value of the second tier (and so on).

If the step boundaries are dynamic (i.e., they are calculated based on something that's only known at billing time), you can change the step boundaries using either of the following methods:

- Use **Step RC Sequence** to indicate the sequence number of the rate component whose value will be multiplied by the step's boundaries to calculate the step boundaries to use at billing time.
- Use **Step Algorithm** to indicate that an algorithm will be called to manipulate the step boundaries (note, this algorithm can manipulate the low and/or high boundaries). If you plan to use this method, you must set up this algorithm in the system. To do this:
  - Create a new algorithm (refer to [Setting Up Algorithms](#)).
  - On this algorithm, reference an Algorithm Type that manipulates a rate component's step boundaries. Click [here](#) to see the algorithm types available for this plug-in spot.

**Warning!** If you dynamically calculate step boundaries and the resultant high AND low values become zero, this signals the system that the rate component should be skipped (i.e., no bill line will be produced for the rate component).

Turn on **Error if No Value** if a bill error should be generated if the UOM / TOU / RQI specified on the rate component was not supplied at billing time.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **GL Distribution** window to define how to book moneys associated with this rate component in the general ledger.

### How To Set Up Apply To Rate Components

There are no special additional fields on this page available for setting up an Apply To rate component. The following information will help you determine how to set up your Apply To rate components.

Select a **Value Type** of **Percent** if the type of charge is a percent of the dollar value of previously calculated rate components. If the amount being charged is an amount per some unit of quantity or measure, select the Value Type of **Unit Rate**. This field will be gray when the rate component references other rate components.

Refer to [Rate Component - Main Information](#) for information about defining the Value Source.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **Cross Reference** window to define the rate components to which the charge will be applied.

Move to the **GL Distribution** window to define how to book moneys associated with this rate component in the general ledger.

### How To Set Up Maximum Charge Rate Components

There are no special additional fields on this page available for setting up a Maximum rate component. The following information will help you determine how to set up your Maximum rate components.

Select a **Value Type** of **Charge**.

Refer to [Rate Component - Main Information](#) for information about defining the Value Source.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **Cross Reference** page to define the rate components against which the maximum charge will be applied.

Move to the **GL Distribution** page to define how any additional revenue associated with the maximum charge should be booked in the general ledger.

### How To Set Up Minimum Charge Rate Components

There are no special additional fields on this page available for setting up a Minimum rate component. The following information will help you determine how to set up your Minimum rate components.

Select a **Value Type** of **Charge**.

Refer to [Rate Component - Main Information](#) for information about defining the Value Source.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **Cross Reference** window to define the rate components against which the minimum charge will be applied.

Move to the **GL Distribution** window to define how any additional revenue associated with the minimum charge should be booked in the general ledger.

### How To Set Up Exact Charge Rate Components

There are no special additional fields on this page available for setting up an Exact Charge rate component. The following information will help you determine how to set up your Exact Charge rate components.

Select a **Value Type** of **Charge**.

Refer to [Rate Component - Main Information](#) for information about defining the Value Source.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **Cross Reference** window to define the rate components against which the exact charge will be applied.

Move to the **GL Distribution** window to define how any additional revenue associated with the minimum charge should be booked in the general ledger.

### How To Set Up Summary Rate Components

Summary rate components use no other fields on this page other than those defined above.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **Cross Reference** window to define the rate components to summarize on this rate component.



### How To Set Up Calculation Algorithm Rate Components

Calculation Algorithm rate components are used when none of the other rate components will provide you with the functionality you need.

First, you must select a **Calc Algorithm**. This calculation algorithm will have the responsibility of producing your bill lines. All rate component options become available when you use this rate component type, based on the assumption that it is generally capable of doing anything you might require.

If you plan to use this method, you must set up this algorithm in the system. This can be done with the following options:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that calculates a rate component. Click [here](#) to see the algorithm types available for this plug-in spot.

**Note.** The calculation algorithm's main purpose is to create bill calculation lines. However, the algorithm may populate other information for the bill, for example, it may add to the RQ or it may overwrite the description on bill.

If you wish to provide your users with the ability to audit the bill lines created by a rate component of this type, you may indicate an **Audit Algorithm**.

All the other fields on the page are optional fields available to you for use by your calculation algorithm.

Turn on **Error if No Value** if a bill error should be generated if the UOM / TOU / RQI specified on the rate component was not supplied at billing time.

Turn on the **Derive RQ** switch when your calculation algorithm will be creating entries in the RQ array for the UOM / TOU / RQI defined on your rate component. In this case, the system will not attempt to locate an entry in the RQ collection matching the rate component's UOM / TOU / RQI before calling your calculation algorithm.

Enter the verbiage to appear on the taxpayer's bill in **Description On Bill** and turn on the **Print** switch. Refer to [How To Use Description on Bill](#) for more information about these fields.

Move to the **GL Distribution** window to define how any additional revenue associated with the minimum charge should be booked in the general ledger.

## Rate Component - Cross Reference

Minimum charge, maximum charge, exact charge, apply-to, and summary rate components reference other rate components. For these types of rate components, open **Main Menu, Rates, Rate Component** and navigate to the **Cross Reference** tab define the other rate components to which it refers.

### Description of Page

**Rate Component** contains a concatenation of basic information about the rate component. This information only appears after the rate component has been added to the database. The adjacent up and down arrows cause the rate component immediately before or after this rate component to be displayed.

Use the **Apply this Rate Component to** collection to specify the other rate component(s) on the Rate Version to which the rate component in question should refer. The list of rate components available for reference is displayed on the right. Use the left arrow to move the desired rate components into the grid. If a rate component appears in the 'apply to' list in error, use the left arrow to remove the rate component.

Refer to [Editable Grid](#) in the system wide standards documentation for more information about adding records to a collection by selecting from a list.

**Note.** An apply to rate component that has a charge type of *unit rate* can only reference RQ and calculation rate components. Apply to rate components with a charge type of *percentage*, along with summary, minimum charge, maximum charge and exact charge rate components can reference all types of rate components including summary.

The **This RC is referenced by** collection indicates what other rate components reference the rate component in question.

## Rate Component - GL Distribution

Open **Main Menu, Rates, Rate Component** and navigate to the **GL Distribution** tab to define each rate component's effect on the general ledger. GL distribution information is required for all rate components except for FCPO and summary rate components.

### Description of Page

**Rate Component** contains a concatenation of basic information about the rate component. This information only appears after the rate component has been added to the database. The adjacent up and down arrows cause the rate component immediately before or after this rate component to be displayed.

If the rate component's charges should be booked to a single GL account regardless of the obligation's obligation type's Revenue Class, enter a **Distribution Code**. If the rate component's charges are booked to a different GL account depending on the obligation's Revenue Class, turn on **Use Revenue Class** and insert the appropriate **Revenue Class** and **Distribution Code** for each that uses the rate.

## Rate Component - Characteristics

Your algorithms and reports may need additional rate component fields that aren't supported in the base-package. If so, you can set up a rate component characteristic for each such field.

**Snapshotting characteristics on bill calculation lines.** The system will automatically copy a rate component's characteristics onto the resultant bill lines for those characteristic types that are usable on bill lines.

Open **Main Menu, Rates, Rate Component** and navigate to the **Characteristics** tab to define characteristic values for your rate component.

### Description of Page

For each characteristic in the collection, indicate the appropriate **Characteristic Type** and **Characteristic Value**.

**Note.** You can only choose characteristic types defined as permissible on the rate schedule record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

## Rate Component - Eligibility

This page is used to define the conditions under which a rate component will be applied. If you don't specify any **Eligibility Criteria Groups**, the system assumes the rate component should be applied (i.e., a rate component is eligible by default).

Refer to [The Big Picture Of Rate Component Eligibility](#) for more information.

Open this page using **Main Menu, Rates, Rate Component** and navigate to the **Eligibility** tab.

### Description of Page

**Rate Component** contains a concatenation of basic information about the rate component. This information only appears after the rate component has been added to the database. The adjacent up and down arrows cause the rate component immediately before or after this rate component to be displayed.

**Warning!** The following information is not intuitive; we strongly recommend that you follow the guidelines under [The Big Picture Of Rate Component Eligibility](#) before attempting to enter this information.

The **Eligibility Criteria Group** scroll contains one entry for each group of eligibility criteria. The following fields may be defined for each group:

- Use **Sort Sequence** to control the relative order in which the group is executed when the system determines if the rate component should be applied (smaller numbers are executed before larger numbers).
- Use **Description** and **Long Description** to describe the criteria group.
- Use **If Group is True** to define what should happen if the eligibility criteria (defined in the following grid) return a value of **True**.
  - Choose **Apply Rate Component** if this rate component should be executed.
  - Choose **Check Next Group** if the next criteria group should be checked.
  - Choose **Skip Rate Component** if this rate component should NOT be executed.
- Use **If Group is False** to define what should happen if the eligibility criteria (defined in the following grid) return a value of **False**.
  - Choose **Apply Rate Component** if this rate component should be executed.
  - Choose **Check Next Group** if the next criteria group should be checked.
  - Choose **Skip Rate Component** if this rate component should NOT be executed.

The grid that follows contains the rate component's eligibility criteria. Think of each row as an "if statement" that can result in the related eligibility group being True or False. For example, you might have a row that indicates the taxpayer is eligible for the rate component if the taxpayer's birth date equates to that of a senior citizen. The following bullets provide a brief description of each field on an eligibility criterion. Please refer to [Defining Logical Criteria](#) for several examples of how this information can be used.

- Use **Sequence** to control the order in which the criteria are checked.
- Use **Criteria Field** to define the field to compare:
  - Choose **Characteristic** if you want to compare a characteristic value that resides on any of the following to a given value (use the adjacent fields to define the object on which the characteristic resides and the characteristic type):
    - The obligation being billed
    - The obligation's account
    - The main person linked to the obligation's account
    - The characteristic location linked to the obligation
    - The obligation's asset (not supported in this release)
  - In addition, you can also use a characteristic value that is derived while the rate is being calculated (characteristic values can be created by RQ rules and many other rate component algorithms)

- Choose **Rate Component Value** if you want to compare the final value of an earlier rate component to a given value. Push the adjacent search button to select the rate component's sequence number.
- Choose **Rate Quantity** if you want to compare the final value of a rate quantity to a given value. Push the adjacent search button to select the rate quantity's **UOM / TOU / RQI**.
- Choose **Algorithm** if you want to compare anything other than the above options. Push the adjacent search button to select the algorithm that is responsible for retrieving the comparison value.
- Use **Criteria Comparison** to define the method of comparison:
  - Choose **Algorithm** if you want an algorithm to perform the comparison and return a value of True, False or Insufficient Data.
  - Choose any other option if you want to compare the **Criteria Field** using a logical operator. The following options are available:
    - Use **>, <, =, >=, <=, <>** (not equal) to compare the **Criteria Field** using standard logical operators. Enter the comparison value in the adjacent field.
    - Use **IN** to compare the **Criteria Field** to a list of values. Each value is separated by a comma. For example, if a field value must equal **1, 3** or **9**, you would enter a comparison value of **1,3,9**.
    - Use **BETWEEN** to compare the **Criteria Field** to a range of values. For example, if a field value must be between **1** and **9**, you would enter a comparison value of **1,9**. Note, the comparison is inclusive of the low and high values.
- The next three fields control whether the related logical criteria cause the eligibility group to be considered True or False:
  - Use **If TRUE** to control what happens if the related logical criterion returns a value of True. You have the options of **Group is true**, **Group is false**, or **Check next condition**. If you indicate **Group is true** or **Group is false**, then the rate component will be **Skipped** or **Applied** based on the values defined above in **If Group is False** and **If Group is True**.
  - Use **If FALSE** to control what happens if the related logical criterion returns a value of False. You have the options of **Group is true**, **Group is false**, or **Check next condition**. If you indicate **Group is true** or **Group is false**, then the rate component will be **Skipped** or **Applied** based on the values defined above in **If Group is False** and **If Group is True**.
  - Use **If INSUFFICIENT DATA** to control what happens if the related logical criterion returns a value of "Insufficient Data". You have the options of **Group is true**, **Group is false**, or **Check next condition**. If you indicate **Group is true** or **Group is false**, then the rate component will be **Skipped** or **Applied** based on the values defined above in **If Group is False** and **If Group is True**.

## Rate Version Merge

---

The following point summarize the many diverse functions available on this transaction:

- A rate version's rate components have unique sequence numbers. This number controls the order in which a rate component is processed. If you need to insert a rate component and there's no space between existing sequence numbers, you can use this transaction to renumber the rate version's rate components.
- You can use this transaction to move a rate component to a different position within a rate version. When a rate version is moved, all references to the rate component are changed to reflect the new sequence number.
- You can use this transaction to delete a rate component. When a rate component is deleted, all references to the rate component are deleted.
- You can use this transaction to copy rate components from other rate versions. For example,
  - You may want to create a rate version that is similar to an existing rate version. Rather than copying all the information from the existing rate version and then removing the inapplicable components, this page may be used to selectively copy specific rate components from the existing rate version to the new rate version.
  - You may have taxpayer-specific rates that are very similar, but still unique. You can use this transaction to build a taxpayer-specific rate. In this scenario, you may choose to create special 'mini' rate versions, one for each of the various options. Then, you could use the rate version merge page to select the components applicable for the new custom rate version.

**The target rate version must exist prior to using this page.** If you are creating a new rate, you must first create the [Rate Schedule](#) and [Rate Version](#) and then navigate to the merge page to copy rate component information.

### Rate: Bakery-1, Monthly, US Dollars

Version: 3-Jan-2002

Type	Description	Price
RQ	School Board Operating	\$300
RQ	Parks and Recreation	\$340
RQ	County Sales Tax Cred	\$370
Sum	Subtotal	N/A
Apply To	Sales tax	6.191%

### Rate: MFCTR-1, Monthly, US Dollars

Version: 1-Mar-2002

Type	Description	Amount
RQ	School Board Operating	\$300
RQ	Parks and Recreation	\$340
RQ	County Sales Tax Credit	\$370
Apply to	City Tax	1.2%
Flat	Fire District Fee	\$40
Sum	Subtotal	N/A
Apply To	Sales tax	6.191%

### Rate: Real Property -1, Monthly, US

Version: 1-Mar-2002

Type	Description	Amount
Apply to	City Tax	1.2%
Flat	Fire District Fee	\$40

**Duplicate versus Merge.** The [Rate Version](#) page itself has [duplication](#) capability. You would duplicate a rate version if you want to a) create a new rate version AND b) populate it with all the rate components from an existing rate version. The [Rate Component Maintenance](#) transaction also allows you to [duplicate](#) a rate component within a rate version.

Open **Main Menu, Rates, Rate Version Merge** to open this page.

#### Description of Page

Select the **Original Rate Version**, which is the target for merging the rate component information.

**Finished rate versions are protected.** Only rate versions that are **In Progress** or **Validated** may be used as the target rate version in the merge functionality. If any changes are made to a **Validated** rate version on this page, the system will move the status back to **In Progress**. If a rate version with a status of **Finished** is selected, it will be displayed, but you will be unable to copy any information to it (and its description will appear in red). If you need to make changes to a **Finished** rate version, navigate to [Rate Version – Main](#) and change the rate version's status to **In Progress**. Refer to [Lifecycle of a Rate Version](#) for more information.

Select the **Merge From Rate Version**, which is your template rate version to copy the rate components from.

**Note.** You may only copy components from one Merge From rate version at a time. If you want to copy components from more than one rate version, select the first Merge From rate version, copy the desired records, Save, and then select the next Merge From rate version.

The left portion of the page displays any existing rate components for the **Original Rate Version**. The right portion of the page displays the existing rate components for the **Merge From Rate Version**.

You can use the **Copy All** button to copy all the rate components from the **Merge From** rate to the **Original** rate. If you use **Copy All**, please be aware that the rate components are added to the end of the **Original** rate version.

Each time you save the changes, the system renumbers the **Original** rate version using the **Start From Sequence Number** and **Increment By**.

The left portion of the page initially displays existing rate component records linked to the original rate version. In the **Merge Type**, you will see the word *Original*, for any of these records. The **Sequence**, **RC Type** and **Description** for each rate component are displayed. In the right portion of the page, the existing records in the merge from rate version are displayed initially.

The topics that follow describe how to perform common maintenance tasks:

### Contents

- [Resequencing Rate Components](#)
- [Removing A Row From A Grid](#)
- [Adding A New Row To A Rate](#)
- [Removing An Uncommitted Row From A Rate](#)
- [Moving Positional Rows Up and Down](#)
- [Rate Component Cross Reference Information](#)

## Resequencing Rate Components

If there are no gaps between two rate components and you need to resequence the rate components:

- Make a change to the **Start From Sequence Number** or **Increment By** (so that the Save button becomes enabled).
- Click the Save button.



## Removing A Row From A Grid

If you wish to remove a record linked to the Original rate version, press the “-” button to the left of the record.

## Adding A New Row To A Rate

You may move any of the rate components from the Merge From rate version to the original rate version by selecting the left arrow adjacent to the desired row. Once a record is moved it will disappear from the Merge From information and appear in the Original information with the word **Merge** in the Merge Type column.

## Removing An Uncommitted Row From A Rate

If you have copied a row across by mistake, you may remove it by clicking on the right arrow adjacent to the appropriate record.

## Moving Positional Rows Up and Down

A rate component's sequence number controls the order in which the rate component should be applied. You may modify the order execution using the up and down arrows.

The sequences need to be renumbered in order to ensure the correct order. The fields **Start from Sequence Number** and **Increment By** are used to define the sequence number to assign to the first rate component and the subsequent values to assign to subsequent rate components.

Refer to [Editable Grid](#) in the system wide standards documentation for more information about adding records to a collection by selecting from a list and repositioning rows within a grid.

## Rate Component Cross Reference Information

All information related to a rate component, including general ledger information, characteristics and eligibility rules, will be copied.

Special logic exists for merged rate components that reference other rate components. This includes the cross reference collection and the eligibility criteria.

If any rate component refers to another rate component in the eligibility criteria, the sequence number of the eligibility criteria is also adjusted accordingly.

**Note.** You will be warned by the system when you copy a rate component that has cross-references to other rate components that are not being copied.

## Rate Check

The Rate Check page allows you to test your rates under various scenarios. The topics in this section describe how to use this transaction.

### Contents

- [Rate Check - Main](#)
- [Rate Check Results - Calc Lines](#)
- [Rate Check Results - RQ Details](#)
- [Rate Check Results - Calculation Details](#)
- [Rate Check Results - Characteristics](#)

## Rate Check - Main

Open **Main Menu, Rates, Rate Check** to enter the parameters required to check a rate.

### Description of Page

**Easiest way to use rate check.** The easiest way to populate the above parameters is to select an existing bill segment on [Bill Segment – Calc Lines](#) and then use the rate schedule context menu to navigate to this transaction. When you do this, the system populates the above fields from the values on the bill segment. After these fields are populated, you can override the defaulted values to check various scenarios.

Indicate the **Rate Schedule** that you want to test. The version(s) used will be those in effect during the start and end dates. Only **validated** and **finished** rate versions will be used.

**Default note.** If **Rate Schedule** is blank and you enter an **Obligation ID**, the obligation's current rate will be defaulted.

While using an **Obligation ID** is not strictly required to check a rate, we strongly recommend finding a representative obligation to check a rate. Why? Because charges in a rate may be affected by values in an obligation. For example, consider the following types of rate components that use obligation-specific values:

- Rate components with rate factors that are subject to the presence of contract riders. In fact, if a rate factor requires a contract rider and you do not specify an **Obligation ID**, rate check assumes that the rider is applicable.
- Rate components with rate factors that are subject to tax exemptions.
- Rate components with rate factors with taxpayer-specific charges.

- Rate components with rate quantity (RQ) rules that use characteristics on obligation, account or the obligation's characteristic location.
- Rate components with RQ rules that use contract quantities.

In other words, if you don't specify an **Obligation ID**, you will not be able to check rates with the above characteristics.

Enter a **Bill Seg ID** if you want to test the rate with actual quantities for a given bill segment. This is useful if you want to compare the actual bill segment results with the results of the same calculation values under a different rate schedule. If you indicate a **Bill Seg ID**, rate check's **Obligation ID** will be populated with the bill segment's obligation ID (and **Obligation ID** will be protected).

**Overriding quantities.** If you enter a **Bill Seg ID**, the bill segment's quantities will be defaulted into the array that follows. You may override the quantities in this array to test various scenarios.

You must indicate a **Revenue Class**. This is needed for applying the appropriate GL distribution code for rate components where the GL account is based on the taxpayer's revenue class.

Enter the **Start Date** and **End Date** that you are testing. If a **Bill Seg ID** is entered, the start and end dates are taken from the bill segment's period, but can be overwritten.

Indicate the **Accounting Date**. If a **Bill Seg ID** is entered, the accounting date on the bill segment is used but can be overwritten.

Enter the quantities that you want to test in the quantity collection. For every **UOM/TOD/RQI** combination, enter a quantity.

The collection of **Characteristic Types** and **Characteristic Values** is used to apply the appropriate rate factor values. If these values are entered, these values will be used to apply the appropriate rate factors (even if you have specified an obligation from which characteristics can be derived). If these values are blank AND there is no obligation ID, then you will receive an error from the rate application indicating which values are needed.

Indicate the **Bill Language** to use for the resulting bill calc line descriptions.

When you have entered everything, click on the **Check Rate** button. This will cause your rate to be applied to the parameters entered above. If the system is successful, you will be transferred to the **Calc Lines** page on which the resulting calculation details can be viewed.

## Rate Check Results - Calc Lines

After you have pressed the **Check Rate** button on the **Main** page, you will be transferred to the **Calc Lines** tab to view the lines that have been calculated by your rate.

### Description of Page

The **Rate Schedule** and its description, the **Revenue Class** and its description and the **Bill Period** are displayed at the top.

More than one Rate Version may have been effective for the Rate Schedule during this bill period. If so, multiple collections of calculation lines will be produced. Use the **Bill Seg Hdr** scroll to view each collection of calculation lines.

For each **Bill Seg Hdr**, the following information will be displayed above the calculation line grid.

<b>Sequence</b>	This identifies which <b>Bill Seg Hdr</b> is being displayed.
<b>Start Date, End Date</b>	This is the period, within the bill period, that the rate version, which produced these lines, was effective.
<b>Amount</b>	This is the total amount calculated by this group of calculation lines.
<b>Desc on Bill</b>	The description on bill for the rate version that produced these lines.
<b>Rate Version</b>	The rate version that was used to produce these lines.

The grid, which displays the bill calculation lines for this rate version, shows one row exists for every calculation involved in this process. This information is very similar to what is displayed on the bill calculation lines page for a bill segment. The following information is displayed in the grid:

- **Sequence** is the system-assigned unique identifier of the calculation detail row.
- **Description on Bill** is the information about the bill line that appears on the taxpayer's bill.
- **Calculated Amount** is the calculated amount associated with the bill line.
- **Calculated Value** is the same value as the **Calculated Amount** except that it displays 5 decimals.
- The **Print** switch controls whether information about this line will print on the taxpayer's bill.
- The **Appears in Summary** switch defines if this line's amount also appears on a summary line. This switch is turned on if the corresponding rate component is summarized on a summary rate component.
- **UOM** is the unit of measure of the rate quantity on the calculation line.
- **TOU** is the time-of-use code of the rate quantity on the calculation line.
- **RQI** is the rate quantity identifier of the rate quantity on the calculation line.
- **Billable Rate Quantity** is the rate quantity on the calculation line.
- **Base Amount** is used by calculation lines (e.g. taxes) that are cross-referenced to other calculation lines and whose value(s), therefore, depend on the amounts calculated by those other lines. The Base Amount shows the total amount derived from the cross-referenced line(s) that the current line then used to calculate its billed amount.

- **Sequence** refers to the sequence number of the rate component on the applicable rate version that was used to calculate the line.
- **Meas Peak Qty** is checked if the UOM on the calculation line is used to measure a peak quantity.
- **Exempt Amount** is the amount of the calculated charge that the taxpayer doesn't have to pay because they are tax exempt. This will only be calculated if you have provided an obligation ID on the main page.
- **Distribution Code** is the distribution code associated with the calculation line. This distribution code is used to build the general ledger details on the bill segment's financial transaction.

## Rate Check Results - RQ Details

After you have pressed the **Check Rate** button on the **Main** page, you can navigate to the **RQ Details** tab to view the rate quantities that supplied to (or derived by) your rate. Please be aware that a rate can derive rate quantities if there are rate quantity rules rate components.

### Description of Page

The **Rate Schedule** and its description, the **Revenue Class** and its description and the **Bill Period** are displayed at the top.

The RQ details grid is a snapshot of the rate quantities amassed from:

- The quantities input on the main page OR the quantities taken from the rate quantity collection for the bill segment, if it was input on the main page.
- The RQ rules defined on the input rate.

The following information is displayed in the grid:

- **UOM** is the unit of measure of the rate quantity.
- **TOU** is the time-of-use code of the rate quantity.
- **RQI** is the rate quantity identifier of the rate quantity.
- **Initial Rate Quantity** is the initial quantity amassed by the system before application of the rate's RQ rules.
- **Billable Rate Quantity** is the rate quantity that will be used by the obligation's rate.

## Rate Check Results - Calculation Details

After you have pressed the **Check Rate** button on the **Main** page, you can navigate to the **Register Read** tab to view the final quantities (Final UOM, Final TOU and Final RQI) that were derived by your rate.

### Description of Page

The **Rate Schedule** and its description, the **Revenue Class** and its description and the **Bill Period** are displayed at the top.

This page displays measured quantities from one of two places:

- From the values input on the main page of Rate Check
- From the quantities collection for the bill segment indicated on the main page of Rate Check

The following information is displayed in the grid:

- **UOM** is the unit of measure of the quantity.
- **TOU** is the time-of-use code of the quantity.
- **RQI** is the rate quantity identifier of the quantity.
- **Quantity** is the initial quantity amassed by the system before application of the rate's register rules.
- **Final UOM** is the unit of measure of the resultant billing quantity.
- **Final TOU** is the time-of-use code of the resultant billing quantity.
- **Final RQI** is the rate quantity identifier of the resultant billing quantity.
- **Final Quantity** is the resultant billing quantity.

## Rate Check Results - Characteristics

After you have pressed the **Check Rate** button on the **Main** page, you can navigate to the **Characteristics** tab to view the final characteristics collection as manipulated by your rate's plug-ins.

### Description of Page

The **Rate Schedule** and its description, the **Revenue Class** and its description and the **Bill Period** are displayed at the top.

The collection of **Characteristic Types** and **Characteristic Values** reflects any changes made to the initial characteristics collection (input on the main tab page) after applying the rate.

## Effective Dates & Proration

---

In this section, we describe how proration applies to different types of rate components. Proration implies assessing charges proportionately.

The term "proration" describes two different issues:

- Prorating a charge whose value changes during a bill period. For example, if a tax rate changes during a bill period and you've indicated that such changes should be prorated, the system prorates the tax change (e.g., 20 days at 5% and 10 days at 6%).
- Prorating charges when the time period being billed is not in sync with the time period in which the charges were defined in the rate. For example, if a rate contains a flat monthly charge and the bill period spans two months, the flat charge must be prorated.

This is a complicated topic as it's possible for many proration issues to exist on a single bill. For example, on a single bill:

- The rate structure can change several times during the bill period (i.e., multiple rate versions are effective).
- The taxing authority changes the tax rate.
- The taxpayer's tax exemption changes.
- The taxpayer-specific charge can change.

### Contents

[Types of Proration](#)  
[Proration Factors](#)  
[Overriding Proration Factors](#)  
[Rate Quantity Proration](#)  
[Rate Component Value Proration](#)  
[Rate Factor Value Proration](#)

## Types of Proration

The following section describes the types of proration performed by applying a rate.

For more information on how prorated charges are presented on the printed bill, refer to [How To Use Description On Bill](#).

### Contents

[Rate Calculation Period Proration](#)  
[Rate Factor Proration](#)  
[An Obligation Rate Changes Are Not Prorated](#)

## Rate Calculation Period Proration

There can be multiple versions of a given rate over time. Each version has an effective date. When the system detects that a rate schedule has multiple versions of a rate in effect during a bill period, it may or may not prorate the charges. You control exactly how the system handles this situation using the rate schedule's proration parameters.

If multiple rate versions are effective during the period and the rate schedule is configured to prorate the rate versions, a given bill segment will contain a set of bill calculation lines for each rate version. You should consider quite carefully how to handle this situation on your printed bills as rate version proration can confuse a taxpayer.

Refer to [Defining Rate Versions](#) for more information about rate versions.

## Rate Factor Proration

A rate contains rate components. When you create rate components, you have two ways to indicate how much to charge:

- You can specify the value directly in a rate component.
- You can have the rate component use a rate factor that contains the value.

Rate factors are effective-dated (i.e., their values may change over time). When the system detects that a rate component's rate factor has multiple values in effect during a period, it may or may not prorate the charges. You control exactly how the system handles this situation using the rate factor's proration parameters.

Refer to [Defining Rate Factors](#) and [Rate Factor Value Proration](#) for more information.

## Contents

[Tax Exemption Proration](#)  
[Contract Value Proration](#)  
[Contract Rider Proration](#)

### Tax Exemption Proration

Rate factors may or may not be eligible for tax exemptions; it's up to the respective taxing authority. Taxpayers who are eligible for tax exemptions will have their tax obligation reduced (or nullified) at billing time.

Whether or not a taxpayer is tax exempt for a given rate factor (and the percent exempt) is defined on the obligation's tax exemption page. The exemption is effective-dated (i.e., it may change over time). If a taxpayer's eligibility for the exemption changes during a period, the system may or may not prorate the charges. You control exactly how the system handles this situation using the rate factor's proration parameters.

### Contract Value Proration

The system allows for types of rate factors that do not have the same value for all taxpayers.



For rate factors whose value is taxpayer-specific, the value is defined on the obligation's contract value page. The contract value is effective-dated (i.e., it may change over time). If a taxpayer's contract value changes during a bill period, the system may or may not prorate the changes. You control exactly how the system handles this situation using the rate factor's proration parameters.

### Contract Rider Proration

Contract riders are used when the charge associated with a rate factor is only applicable to some taxpayers for some period of time.

A taxpayer's contract rider applicability period is defined on the obligation's contract rider page. A rider's applicability may change during a taxpayer's bill period. You control exactly what the system does when a rider's applicability changes during a bill period using the rate factor's proration parameters.

### An Obligation Rate Changes Are Not Prorated

An obligation's rate is effective-dated (i.e., it may change over time). When the system calculates an obligation's bill segment when multiple rates are effective during the bill period, it will use a single rate from the obligation's rate history. The system chooses the rate as per the obligation's obligation type's rate selection date.

Refer to [Obligation Type - Rate](#) for more information.

## Proration Factors

The purpose of rate components is to generate bill calculation lines. When rate application builds bill calculation lines, it may also prorate the charges.

There are multiple types of proration performed by rate application. Rate quantities, step boundaries, rate values, and rate factor values may all be prorated as part of applying a rate.

We need to establish some terms in order to understand the calculations.

### Contents

- [Normal Days](#)
- [Calculation Period Factor](#)
- [RF Value Period Factor](#)

### Normal Days

As mentioned previously, the rate schedule references a frequency that specifies the number of times per year that obligations referencing the rate are expected to bill. The following formula dictates a rate frequency's normal days:  $365 \text{ days} / \text{number of periods per year}$ .

Example:

- For a monthly frequency (12 periods per year),  $365 / 12 = 30$  days.
- For a quarterly frequency (4 periods per year),  $365 / 4 = 91$  days.

Refer to [Defining Frequency Codes](#) for more information.

### Calculation Period Factor

The calculation period is the date range where a rate version is effective during the billable period. For each rate version that is effective during the billable period, a bill segment calculation period is created. A set of bill calculation lines is created for each calculation period.

A calculation period factor is determined for each calculation period. The calculation period factor is the ratio of calculation period days to normal days. If the billable period factor is 1 (not applicable), then the calculation period factor is the ratio of calculation period days to consumption period days.

### RF Value Period Factor

The rate factor value period is the date range where a prorable rate factor value is effective during the calculation period.

When a rate factor value changes during the calculation period, the results is that multiple bill calculation lines are generated for the same rate component.

The ratio of the number of days within the calculation period that a distinct value is effective to the number of days in the calculation period determines the RF value period factor. When the rate factor is set up not to prorate, the RF value period factor is 1 (not applicable).

Example:

If the state tax rate changes from 6% to 6.5% on April 16 and the calculation period is April 1 to April 30, there are two rate factor value periods.

- Rate factor value period April 1 to April 15, RF value period factor = 15 days / 30 days = 0.5.
- Rate factor value period April 16 to April 30, RF value period factor = 15 days / 30 days = 0.5.

Refer to [Defining Rate Factors](#) for more information on specifying whether the rate factor allows proration.

## Overriding Proration Factors

If the system logic for determining proration factors does not satisfy your requirements, you may plug in an override proration algorithm to calculate the proration factors as required by your implementation. The override proration algorithm must be specified on the [Installation](#) record.

The system is delivered with an algorithm type that sets the proration factors to 1 (not applicable) when a predefined characteristic is specified on a rate component. Refer to [OVPF-NOPROR](#) for more information.

## Rate Quantity Proration

### Contents

[Step Proration](#)

[Step Multipliers](#)

### Step Proration

If a rate component's charges are based on stepped charges then the step boundaries are also prorated (as long as the rate component is not marked as measures peak). Step boundaries are multiplied by the calculation period factor to determine the adjusted steps boundaries.

### Step Multipliers

If the rate component specifies that the steps be multiplied by the result value of another rate component, step proration is not performed. It is assumed that the result value from the referenced rate component has already been prorated, as the referenced step rate component is subject to [Rate Component Value Proration](#).

## Rate Component Value Proration

Except for summary type rate components (which are not prorated at all), a value is specified on the rate component. The value may be a unit rate, percentage or charge. The value may be specified directly on the rate component, derived from a rate factor, derived from another rate component or calculated by an algorithm.

Regardless of where the rate component value comes from, it will be multiplied by the calculation period factor when determining the charge on a bill calculation line. Exceptions to this rule are:

- Unit rate values where the rate component's measures peak attribute is not checked.
- Percentage values.
- Rate component values derived from another rate component.

Even though the rate component values are not prorated for unit rate and percentage value types, if these rate values change during the calculation period due to a rate factor value change, rate factor value proration still applies.

Values derived from another rate component are not subject to rate value proration, as rate component value proration was already performed on the referenced rate component. Again, if the rate factor value changed on the referenced rate component, rate factor value proration still applies.

For more information on rate factor value proration, refer to <a href="#">Rate Factor Value Proration</a> .
---

## Rate Factor Value Proration

Rate factor value proration behaves the same way any time a rate factor value changes within a calculation period. This could be due to a change in a tax rate, a contract rider becoming effective, a tax exemption expiring, etc.

Multiple bill calculation lines are generated for a given rate component when a proratable rate factor value changes during the calculation period.

Example:

If the state tax rate changes from 6% to 6.5% on April 16, and the calculation period is April 1 to April 30.

- The first RF value factor period April 1 to April 15, so RF value period factor is 0.5.
- The value used in the first bill calculation line for state tax is  $6\% \times 0.5 = 3\%$ .
- The second rate factor value period is April 16 to April 30, so the RF value period factor is 0.5.
- The value used in the second bill calculation line for state tax is  $6.5\% \times 0.5 = 3.25\%$ .

## How To Add A New Rate Quantity Rule Algorithm

**Technical Information Below!** This section is meant for technicians only!

If the RQ rule algorithms that are supplied with the base package are not sufficient for your rating requirements, you will have to write a new RQ rule algorithm. This section describes how to do this. After you have followed the steps outlined below, you can then set up the control tables necessary to invoke this algorithm as described under [Setting Up Rate Quantity Rules](#).

Follow the following steps to define a new rate quantity (RQ) rule:

- Assign a two-character identifier to the RQ rule. This identifier should begin with an **X** or **Y**. For example, you could give your first RQ rule the identifier **X1**, the second **X2**, etc.
- Create an algorithm type for your new RQ rule. This algorithm type exists to define the types of parameters supplied to your RQ rule. Refer to [Base Package RQ Rules](#) for examples. This algorithm type must be identified by the two-digit identifier allocated above.

While RQ rules require an algorithm type, they do NOT require an algorithm. Why? Because you set up an RQ rule to define the values of the parameters passed to your program (and therefore you don't need an algorithm as algorithms exist to define the values of the parameters passed to a program).

- Go to [Look Up Options](#) and add a new lookup value for the field ***SQ\_RULE\_FLG***. This should have a Field Value equal to the two-digit identifier allocated above.
- Write your new program.

# Pay Plans

Pay plans are typically used to satisfy obligations in which the taxpayer cannot satisfy the total obligation with one lump payment. Payments received are used to satisfy tax, penalty, and interest. Pay plans are typically negotiated between the tax authority and taxpayer. The negotiated payment amount and specific date intervals of the pay plan include such factors as total obligation amount and the taxpayer's ability to pay. Examples in which a payment plan should be used include:

- After an audit of income, it is determined that the taxpayer incorrectly calculated their tax and owes additional tax to the tax authority. If they do not have enough money to satisfy the obligation at the time of the audit decision, they can arrange a series of payments to the tax authority to satisfy their obligation.
- Property values rose by an unexpected rate. This increased property bills and if a taxpayer could not pay the tax bill on the new assessed value of their house they could set up a pay plan to pay off the amount they owe over a six-month period.

Pay plans can cover one or more obligations linked to any account of the taxpayer. You can have multiple active pay plans for an account, but any given obligation can only be covered by a single pay plan at any point in time.

## Contents

[The Big Picture Of Pay Plans](#)  
[Maintaining Pay Plans](#)  
[How To](#)

## The Big Picture Of Pay Plans

---

The following topics provide background information about pay plans.

## Contents

[Pay Plans Are Obligations](#)  
[Which Obligations Are Covered By A Pay Plan?](#)  
[Navigating To The Pay Plan Maintenance Page](#)  
[Recommending Scheduled Payments](#)

## Pay Plans Are Obligations

Pay plans are initiated by creating a 'pay plan' type of obligation for a taxpayer. The pay plan and its obligation type maintain the details of the pay plan, such as recommendation rules, covered obligations, and scheduled payments.

Pay plans are just like other obligations in that collections activity may be used to encourage the taxpayer to pay.

Pay plans differ from other obligations in the following ways:

- Pay plans have a special role of **Pay Plan**.
- Pay plans have scheduled payments.
- Pay plans maintain a list of covered obligations.

- Pay plans are started and maintained on a separate transaction (refer to [Pay Plan - Main](#) for more information).

## Which Obligations Are Covered By A Pay Plan?

A pay plan can apply towards one or more of the taxpayer's obligations whose obligation type is flagged as **Eligible** (for Pay Plan). Obligations can be added to or removed from a pay plan at any time. Additionally, an obligation can be linked to more than one pay plan as long as the pay plans to which it is linked do not have overlapping start and end dates.

A list of the obligations covered by a pay plan is maintained with the pay plan. This list is used during payment distribution to determine the priority of distribution.

## Navigating To The Pay Plan Maintenance Page

Any of the following methods can be used to navigate to the Pay Plan Maintenance transaction:

- An alert in the [Alert Zone](#) highlights if a pay plan exists for the current account. You are transferred to the pay plan transaction if you drill down on the alert.
- A user can setup a [favorite link](#) associated with the pay plan transaction.
- Via any of the following menus:
  - Account context menu
  - Obligation context menu
  - Main, Compliance menu

## Recommending Scheduled Payments

When creating a pay plan, you must select a recommendation rule. The recommendation rule establishes the amount to be paid and the dates on which the payments are due. In other words, the recommendation rule creates a recommended payment schedule.

The recommendation rules that you can use are specific to your implementation. The recommendation rule that you select can have default values for its parameters. If your implementation allows, you can manually override one or more of these default values. For example, you may be able to specify the period covered by the pay plan and the day of the month on which the payments are due.

If your implementation allows, you can manually modify the recommended payment schedule to suit the needs of a particular taxpayer.

Refer to <a href="#">Designing Recommendation Rules</a> for more information.
---

## Maintaining Pay Plans

Pay plans are started and maintained on a different transaction than other obligations. The topics in this section explain how to start and maintain pay plans.

**Contents**

- [Pay Plan - Main](#)
- [Pay Plan - History](#)
- [Pay Plan - Characteristics](#)

## Pay Plan - Main

The Main page contains basic information about the pay plan. Open **Compliance, Pay Plan** to maintain this information.

**Description of Page**

**Pay Plan** information and **Pay Plan ID** are displayed on each page. These values only appear after the obligation is created. The **Pay Plan ID** is a system assigned random number that stays with the obligation for life. The **Pay Plan** information is a concatenation of important details about the obligation and its account.

The **Primary Account ID** is identification of the account that is financially responsible for the pay plan.

The **Primary Taxpayer ID** is identification of the taxpayer that is financially responsible for the pay plan.

Indicate the **Division** and **Obligation Type**. These fields control aspects of the pay plan's behavior, such as events that occur when a pay plan is activated or stopped. These fields are gray when the status is not **Pending Start**.

For more information about pay plan obligation types, refer to [Defining Pay Plan Options](#).

**Recommendation Rule** is used to create a recommended payment amount and payment schedule. The obligation type controls which recommendation rules are allowed for the pay plan.

If **Scheduled Payment Auto Pay** is **Included in Auto Pay** and the account is set up for automatic payment, the scheduled payments are covered by auto pay. If **Excluded from Auto Pay**, the scheduled payments are excluded from auto pay if it is set up for the account.

The **Start Date** defines when the financial relationship begins. The **End Date** is a display only field that indicates when the financial relationship terminates.

The **Recommendation Rule Parameter Values** grid specifies the parameters and their values to use for the selected recommendation rule:

- **Parameter** is the name of the parameter from the recommendation rule algorithm type.
- **Value** is the value used for the parameter. If the recommendation rule allows, you can override the default values by specifying your own.

**Note.** The recommendation rule's controls the number and type of parameters. The recommendation rule controls the default values and whether they can be overridden.



The **Covered Obligations** grid lists the obligations that are covered by the pay plan. The list of covered obligations would depend on the business situation and the rules of the specific tax authority. Some tax authorities might allow for obligations across different tax types to be listed on the same pay plan while others may not. Rules regarding open cases (audit, bankruptcy) would also limit the eligibility of obligations. You must create the list of covered obligations before the recommendation rule can recommend a payment amount and list of scheduled payments. The grid displays the **Obligation ID**, **Obligation Information**, **Account ID**, **Current Balance** and **Payoff Balance** for covered obligations.

The **Scheduled Payments** grid displays the scheduled payments created by the recommendation rule (after you click the Recommend button). You can manually create or modify the **Scheduled Date** and or **Scheduled Amount** of any scheduled payment in the list. The **Total of Scheduled Payments** shows the total of all scheduled payments in the list. The **Status** of each scheduled payment is also displayed.

In the Action grid:

- Click **Recommend** to create a recommended payment schedule based on the recommendation rule.
- Click **Break** to manually stop the pay plan. This executes any break pay plan algorithm(s) defined for the obligation type, and changes the status of the obligation to **pending stop**.
- Click **Cancel** to cancel the pay plan. This executes any cancel algorithm(s) defined for the obligation type, and changes the status of the obligation to **canceled**.

## Pay Plan - History

The Financial History page displays the pay plan's historical financial transactions. Open **Compliance**, **Pay Plan** and navigate to the **History** tab to view this information.

### Description of Page

**Pay Plan** information and **Pay Plan ID** are displayed on each page. These values only appear after the pay plan is created. The Pay Plan ID is a system assigned random number that stays with the obligation for life. The Pay Plan information is a concatenation of important details about the obligation and its account.

The **Account ID** identifies who is financially responsible for the obligation.

The grid shows the pay plan's financial history in reverse chronological order. This includes financial transactions that directly affect the balance of the pay plan obligation, such as fees. The following fields are displayed:

- The **Effective Date** is used to compute how many days old the debt is.
- **Financial Transaction Type** displays the type of financial transaction except for adjustments. For adjustments, the adjustment type's description is shown.
- **Current Amount** is the effect of the financial transaction on the obligation's current balance.
- **Current Balance** is the amount owed by the taxpayer at the time of the transaction.

## Pay Plan - Characteristics

The characteristics page contains information that describes miscellaneous information about the pay plan obligation. Use **Compliance**, **Pay Plan**, **Characteristics** to open this page.

## Description of Page

**Note.** You can only choose characteristic types defined as permissible on the obligation record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

The following fields display:

<b>Effective Date</b>	Indicate the date on which the characteristic value becomes effective. Note, the effective date defaults to the pay plan start date.
<b>Characteristic Type</b>	Indicate the type of characteristic.
<b>Characteristic Value</b>	Indicate the value of the characteristic.

## How To

---

The topics in this section describe how to perform common pay plan functions.

### Contents

- [How To Create A Pay Plan](#)
- [How To Activate A Pay Plan](#)
- [How To Break Pay Plans](#)
- [How To Cancel Pay Plans](#)
- [How To Set Up Automatic Payment For A Pay Plan](#)

## How To Create A Pay Plan

To set up a new pay plan:

- Navigate to the Pay Plan maintenance page.
- If the account is not already specified, select the **Account** for which you want to create a pay plan
- Define the **Division** and **Obligation Type** to use for the pay plan.
- If necessary, change the **Start Date** (the current date is defaulted).
- Select the **Recommendation Rule** to use and modify any of the default parameters to meet the needs of the taxpayer.
- Verify the list of **Covered Obligations**.
- Click **Recommend**.
- On the pop-up dialog, specify the **Schedule Start Date** and click **Recommend**.
- Make any necessary changes to the scheduled payments grid and click **Save**.

## How To Activate A Pay Plan

The pay plan is created in a **pending start** status. In order for the pay plan to become effective, it must be activated. Some tax agency's will require an approval or review before a pay plan can be activated. There are a couple of ways a pay plan can be activated:

- From the Pay Plan maintenance page:
  - If the account with the pay plan you want to activate is not specified, select the **Account** of the pay plan
  - If the account has more than one pay plan, select the pay plan you want to activate.
  - From the pay plan context menu, navigate to the Obligation page.
  - Click **Activate**.
- From the Obligation maintenance page:
  - If the pay plan you want to activate is not specified, select the pay plan using the obligation search.
  - Click **Activate**.

## How To Break Pay Plans

An **active** pay plan may need to be stopped due to the taxpayer breaking the terms of the agreement. To stop a pay plan:

- Navigate to the Pay Plan maintenance page.
- If the account with the pay plan you want to stop is not specified, select the **Account** of the pay plan
- If the account has more than one pay plan, select the pay plan you want to stop.
- Click **Break**.

## How To Cancel Pay Plans

A pay plan may need to be cancelled for various reasons: the pay plan was created in error or the taxpayer has indicated additional hardship and a new pay plan needs to be negotiated. A **pending start**, **active**, or **pending stop** pay plans may be cancelled. To cancel a pay plan:

- Navigate to the Pay Plan maintenance page.
- If the account with the pay plan you want to stop is not specified, select the **Account** of the pay plan
- If the account has more than one pay plan, select the pay plan you want to cancel.
- Click **Cancel**.

## How To Set Up Automatic Payment For A Pay Plan

If a taxpayer wants to pay their pay plan scheduled payments automatically:

- Set up the account for automatic payment (as described under [How To Set Up A Taxpayer To Pay Automatically](#)).

- Navigate to the Pay Plan Maintenance page and select the pay plan for the account.
- Set the **Scheduled Payment Auto Pay** field to indicate that the scheduled payments are *Included in Auto Pay*.

# Case Management

**Separate module.** Please note that the Case Management functionality is associated with a separate **Case Management** module. If this module is not applicable to your business you may turn it off. Refer to [Turn Off A Function Module](#) for more information.

Your organization can use case management functionality to manage a variety of situations, such as:

- A formal appeal
- A bankruptcy
- A collection case
- Refund voucher processing
- A taxpayer's request for literature
- Returns and tax form processing

**Your implementation controls how your cases behave.** Since virtually all aspects of case management functionality are controlled by your implementation, you can use it to handle a myriad of business requirements. Refer to [Defining Case Management Options](#) for the details.

## Contents

[Background Topics](#)  
[Maintaining Cases](#)

## Background Topics

---

The topics in this section provide background information about case management functionality.

## Contents

[Case Type](#)  
[Creating Cases](#)  
[Log Information](#)  
[Alerts](#)  
[ToDo's and Cases](#)  
[Reports](#)

## Case Type

A case is controlled by the case type that it references. Some aspects of the case that are controlled by the case type include:

- The [lifecycle](#) of the case
- The [applicability](#) of a person, account, and/or location on the case
- Which [fields are required](#) during the various stages of a case's life

- Which users can [access cases](#) of a given type
- Additional processing that should take place when a case enters or leaves a given state

Refer to [Case Type Controls Everything](#) for a description of the business rules that you control when you set up your case types.

## Creating Cases

To create a case, open the [case management](#) page in add mode, select the appropriate case type, fill in the other necessary fields, and click save. Oracle Enterprise Taxation Management also includes scripts that can help create cases. Refer to [Scripts and Cases](#) for more information.

You can also develop plug-ins, user exits, batch processes, etc so that the system will create cases when specific events take place. For example, you can set the system to create a case when an overpayment on an obligation is deemed refundable or when refund request cases need to be grouped into vouchers for approval.

Please refer to [Maintaining Cases](#) for a detailed description of the case page and field descriptions.

## Log Information

A case log contains an entry for every recorded event during the lifecycle of the case. There are two types of log entries:

- **Automatic entries.** The system automatically creates an entry in the log when a case is created or there is a [status change](#). Users cannot modify or delete these log entries.
- **Manual entries.** Users can add manual entries to record significant events at their discretion.

Refer to [Case - Main](#) for the page used to view the log.

Refer to [Case Statistics for a Given Status](#) for information about a report that uses the log to provide productivity statistics.

## Alerts

The [alert zone](#) will highlight if the person / account / location in context have cases in a state that is configured as “alertable” if you plug-in the appropriate algorithm on the installation record. Refer to [Alert Info Is Controlled By An Installation Plug-In](#) for the details.

## ToDo's and Cases

You can configure the system to inform users of cases that require their attention. Refer to [ToDo's and Cases](#) for the details.

## Reports

Several reports summarize case information. Refer to the [Reports](#) chapter for the details.

## Maintaining Cases

---

Since an implementation can configure virtually every aspect of a case, the look and feel of the case management page are dynamic based on how [case types](#) have been configured.

### Contents

- [Case - Main](#)
- [Case - Case Portal](#)
- [Case - Log](#)

## Case - Main

Open the case management page by selecting **Taxpayer Information, Case** from the main menu.

### Main Information

**Case Info** contains a concatenation of important information about the case. **Case ID** is the system-assigned unique identifier of the case. These values only appear after the case is added to the database.

**Formatting may be performed by a plug-in.** Refer to [Case Info May Be Formatted By An Algorithm](#) for more information.

**Case Type** defines the type of case. This field is protected after the case has been added. Refer to [Case Type Controls Everything](#) for more information.

**Status** shows the current state of the case. Refer to [Case Lifecycle](#) for more information.

If the case has any ToDo entries associated with it, a hyperlink will appear summarizing the number and status of the entries. You can select the link to view these entries. Refer to [ToDo's and Cases](#) for more information.

**Date / Time Opened** shows when the case was created. If the case is in a final state, the **Date / Time Closed** will also be shown. Refer to [One Initial State and Multiple Final States](#) for more information.

**Script** references the business process assistant script that can help you work on the case. You can click on the hyperlink to execute this script. This field is hidden if no script is associated with this case's current Status. Refer to [Scripts and Cases](#) for more information.

The **Actions** section contains buttons you can use to action the case. This area is suppressed if no actions can be performed for the case given its current status. A button will be gray if you do not have security rights to the action. Refer to [Buttons Are Used To Transition A Case](#) for more information.

When the user adds a new case or changes the state of a case manually the system attempts to auto-transition the case to subsequent statuses as necessary. If auto-transition rules apply to the new state (and to subsequent ones) they would be executed right away. If the case has been automatically transitioned to its current state the following message area appears right after the action area. If an error is encountered in the process a corresponding error message is displayed. Refer to [Automatic Transition Rules](#) for more information.

**Note.** This section only appears when a case has just been automatically transitioned to its current state online. It only remains visible until the data for the case is refreshed.

Use the **Comment** area to describe anything interesting or unusual about the case.

**Person** references the person associated with this case. This field is suppressed if the Case Type indicates that a person is not allowed on cases of this type. This field is protected if the case is closed (i.e., if its status is defined as being final). This value is defaulted to the Person currently displayed in the Dashboard.

**Account** references the account associated with this case. This field is suppressed if the Case Type indicates that an account is not allowed on cases of this type. This field is protected if the case is closed (i.e., if its status is defined as being final). This value is defaulted to the Account currently displayed in the Dashboard.

**Location** references the location associated with this case. This field is suppressed if the Case Type indicates that a location is not allowed on cases of this type. This field is protected if the case is closed (i.e., if its status is defined as being final). This value is defaulted to the Location currently displayed in the Dashboard.

**Responsible User** references the user who has overall responsibility for the case. This field is suppressed if the Case Type indicates that a responsible user is not allowed on cases of this type. This field is protected if the case is closed (i.e., if its status is defined as being final).

The **Contact Information** area is used to define how to contact the individual who initiated the case. These fields are suppressed if contact information is not allowed on cases of this type. The fields cannot be modified if the case is in a closed status. The following information can be defined.

- **Contact Person** references the person who should be contacted. This defaults to the Person defined above.
- **Preferred Contact Method** indicates how the person prefers to be contacted. The following options are available:
  - **Email.** If this method is selected, the Contact Person's email address appears adjacent.
  - **Fax.** If this method is selected, the Contact Person's fax number appears (the system knows which of a [person's phone numbers](#) is a fax number by the phone type).
  - **Not Applicable.** If the person doesn't want to be contacted (or the contact method is unusual), use this option. You might want to consider describing the situation in the **Contact Instructions**.
  - **Phone.** If this method is selected, the Contact Person's phone number appears.
  - **Postal.** If this method is selected, the Contact Person's address appears.

**Contact Instructions** indicates any special instructions indicated for when or how to return a call.

**Callback Phone Type**, **Callback Phone Number**, and **Extension** indicate the actual phone number and phone type that should be used to call.

### Characteristics

The **Characteristics** grid contains fields that contain important information about the case. This grid is suppressed if the Case Type does not use characteristics. This grid is protected if the case is closed (i.e., if its status is defined as being final).



You can only choose **Characteristic Types** defined as permissible for this case type. Refer to [Additional Information](#) for more information. If you attempt to use an **Action** to transition a case into a status that requires additional information, a pop-up will prompt you for the additional fields. Refer to [Required Fields Before A Case Enters A State](#) for more information.

## Case - Case Portal

To view additional information associated with a case, open by selecting **Taxpayer Information, Case** from the main menu and then the **Case Portal** tab.

### General Information

**Case ID** is the system-assigned unique identifier of the case.

**Case Info** contains a concatenation of important information about the case.

### Object Portal

When information associated with the case has been stored in the case Character Large Object (**CLOB**) field (such as the data associated with a collections case), the **Object Portal** displays this information as-is. You cannot modify the case CLOB field data from this page, but case type algorithms can be configured to work with this data. For information on mapping data into CLOB fields, refer to the Business Object tips document **Schema Nodes and Attributes**.

## Case - Log

To view the log entries associated with a case, open by selecting **Taxpayer Information, Case** from the main menu and then the **Log** tab.

### Description of Page

The **Log** grid displays log entries (in reverse chronological order) that audit the progress of the case. Please note the following about the entries in this grid:

- The system automatically creates log entries when a case is created, every time its status changes, or whenever an error occurs during the transition of a case. Algorithms plugged into a case type state can also be configured to create log entries. You cannot modify or delete the log entries.
- You can manually add a log entry by pressing the + button and enter the **Details**. You cannot modify or delete this information after saving it.
- The following information appears in the grid:
  - **Log Date/Time** contains the date and time the log entry was created.
  - **Details** contains user-specified or system-generated information about the log entry.
  - **Related Object** is populated on log entries that were created to record the creation of some other object. For example, if a case plug-in creates a customer contact, the related object contains information about the customer contact. Please note that if the object description is shown in blue, you can click on the object's description to drill down to the object.
  - **Log User** contains the user who caused the log entry to be created.
  - **Log Type** indicates how the log entry was created. The possible values are:

- **Created.** Only one entry per case will be of this type. This entry is automatically created by the system when the case is created.
- **User Details.** A user creates this type of entry manually.
- **Status Transition.** This type of entry is created by the system each time the case's status is changed (as a result of pressing one of the Action buttons).
- **System.** An algorithm on the case type state may be designed/configured to create this type of entry when it does something auditable. For example, an algorithm that creates a customer contact could insert a log entry to show that it did this (it will also populate the FK to the customer contact, which will appear in the **Related Object** column).
- **Transition Error.** The background status transition process creates an entry of this type if an error occurs when automatically updating a case to a subsequent status (as directed by either an Enter Status algorithm or an Auto-Transition algorithm). An entry of this type lists the case states involved with the failed transition, and the algorithm in which the error was generated.
- **Exception.** The background status transition process creates an entry of this type under the same circumstances as a Transition Error. A log entry of this type indicates why the background transition failed.

**Note.** The values for this field are customizable using the Lookup table. This field name is CASE\_LOG\_TYPE\_FLG.

# Overdue Financial Obligations

The system periodically monitors how much your taxpayers owe to ensure they haven't violated your overdue rules. When a violation is detected, the system creates an overdue process. The overdue process contains the events meant to prod the taxpayer to pay (e.g., letters, To Do entries, write-off outstanding debt, etc.). This section describes how to manage your overdue debt processing.

## Contents

- [Overdue Processing Background Information](#)
- [Overdue Process Maintenance](#)
- [Maintaining Collection Cases](#)
- [Collection Referral](#)

## Overdue Processing Background Information

In the section, [The Big Picture Of Overdue Processes](#), we describe how overdue processing works and how to set up the control tables that automate most functions. You will find that many of your questions regarding when and how overdue processes are created and canceled are described in this section.

## Overdue Process Maintenance

An overdue process is a series of events (e.g., letters, ToDo entries) meant to encourage an account to pay its delinquent debt. The topics in this section describe the pages on which overdue processes are maintained.

Refer to [The Big Picture Of Overdue Processes](#) for background information.

## Contents

- [Overdue Process - Main](#)
- [Overdue Process - Events](#)
- [Overdue Process - Log](#)

## Overdue Process - Main

The Main page contains core overdue process information. Open this page using **Compliance, Overdue Process** from the main menu.

## Description of Page

Refer to [How To Perform Common Overdue Process Functions](#) for more instructions describing how to use this page.

**Overdue Process** displays a concatenation of important information about the process. **Overdue Process ID** is the system-assigned unique identifier of the process. These values only appear after the overdue process is added to the database.

**Summary information may be overridden.** Refer to [Overdue Process Information Is Overridable](#) for how your implementation can override the summary information that appears throughout the system.

**Account ID** identifies the overdue process's account.

**Multiple overdue processes may be linked to an account.** It's important to be aware that it's possible for multiple, active overdue processes to be linked to an account. The [Alert zone](#) will contain a summary of the account's overdue processes.

**Status** defines the state of the overdue process. The following values may exist:

- **Active, Events Pending.** Overdue processes are initially created in this state. An overdue process remains in this state until there are no **Pending** or **Waiting** events.
- **Inactive, Canceled by User.** An overdue process will exist in this state when it's been manually canceled by a user. Navigate to the **Log** tab to see when this happened and who did it.
- **Inactive, Canceled by System.** An overdue process will exist in this state when it's been canceled by the system (typically because the overdue obligations were satisfied). Navigate to the **Log** tab to see when this happened.
- **Inactive, Completed.** An overdue process will exist in this state when the system completes its last event. Navigate to the **Log** tab to see when this happened.

The **Cancel** button appears if the overdue process is **Active, Events Pending** and the process's **Overdue Process Template** has a **Cancel Logic** plug-in. Click this button to [cancel the process](#).

The **Trigger Events** button appears if the overdue process is **Active, Events Pending**. Click this button to activate all pending events that are [ready for activation](#).

**If you can't wait for the Overdue Event Manager to run.** The [Overdue Event Manager](#) is a background process that activates events on their trigger date. If you don't want to wait for this process to run, you can click the **Trigger Events** button on the **Main** tab to activate the process's events (that are ready for activation).

**Overdue Process Template** defines the template that was used to [create the overdue process's events](#). You can override these events on the **Events** page. This field is unprotected when the process is **Active** and all events are in either the **Pending** or **Canceled** states.

**Changing the template.** If you change the template, the system will delete the events and replace them with the new template's events.

**Start Date/Time** defines the start date/time of the overdue process. This field is protected after the overdue process is saved on the database. This field is used to derive the trigger date of some overdue events when the process is first created. Subsequent changes to the start date change the trigger dates on the events accordingly.

**Inactive Date/Time** is the date and time that the overdue process became **Inactive**. This field is hidden if the process is **Active**. You can see more details about when and how the inactivation occurred by navigating to the **Log** tab.

Enter any **Comments** about the overdue process. This field is protected when the overdue process is **Inactive**.

The **Collecting On** grid holds the [overdue objects](#). For example, if this process manages overdue bills, the grid contains a list of the bills being collected on by this process. This grid is unprotected when the process is **Active**. The following points describe the columns that appear in the grid:

- Press the + button to add a new overdue object to the process.
- Press the - button to remove an overdue object from the process.
- The next column contains the unique identifier of the overdue object. For example, if this process manages overdue bills, this column will contain the bill ID. The type of object that's specified in this column is control by the [overdue process template](#).
- **Original Amount** contains the [original amount](#) of the overdue object.
- **Unpaid Amount** contains the [unpaid amount](#) of the overdue object.

The [Trees](#) at the bottom of the page show a variety of information about the overdue process including its events. You can click on hyperlinked tree nodes to navigate to the page on which the related object is maintained.

## Overdue Process - Events

This page contains the activities that are performed to persuade the taxpayer to pay the outstanding debt. Open this page using **Compliance, Overdue Process** and then navigate to the **Events** tab.

### Description of Page

Refer to [How To Perform Common Overdue Process Functions](#) for more instructions describing how to use this page.

Refer to the first tab for a description of **Overdue Process** and **Overdue Process ID**.

The **Overdue Events** scroll contains the process's overdue events. Refer to [Overdue Processes Are Created From Templates](#) for information about how the system defaults a process's events from its template.

- Note, all information in the scroll is protected if the event's Status is **Complete** or **Canceled**.
- **Event Sequence** is the unique identifier of the event.
- **Status** defines the [state of the event](#).
- The **Cancel** button appears if the event is **Pending** or **Waiting**, and the event's **Overdue Event Type** has a **Cancel Logic** plug-in. Click this button to cancel the event.
- The **Overdue Event Type** defines the event's activity (e.g., Email sent, a ToDo entry is generated, a letter is sent).

- The remaining fields work in unison:
  - If **Dependent on Other Events** is turned on, the event can only be triggered after the events specified in the **Event Dependencies** grid are all **Complete** or **Canceled**. You use the **Days After** field to define when this event is activated. For example, if you enter 5, this event will be activated 5 days after all of the dependent events are completed / cancelled. If you enter 0, this event will be activated immediately after the dependent events are completed / cancelled. Refer to [Calendar vs. Work Days](#) for a descriptions of how days are counted.  
  
If all dependent events are already **Completed** or **Canceled**, you cannot specify **Days After**. Rather, you must enter the desired activation date in the **Trigger Date** field.
  - If **Dependent on Other Events** is turned off, enter the desired activation date in the **Trigger Date** field. If this field is off, the **Days After** field and the **Event Dependencies** grid are hidden.

If you can't wait for the Overdue Event Manager to run. The [Overdue Event Manager](#) is a background process that activates events on their trigger date. If you don't want to wait for this process to run, you can click the **Trigger Events** button to activate the process's events (that are ready for activation).

## Overdue Process - Log

This page contains [log entries](#) that highlight significant events in the process's life. Open this page using **Compliance, Overdue Process** and then navigate to the **Events** tab.

### Description of Page

Please note the following about the entries in this grid:

- The system automatically creates log entries when significant events occur. Refer to [Overdue Log](#) for more information.
- You can manually add a log entry to an Active process by pressing the + button and enter the **Details**. You cannot modify or delete this information after saving it.
- The following information appears in the grid:
  - **Date/Time** contains the date and time the log entry was created.
  - **Details** contain the user-specified or system-generated information about the log entry.
  - **Related Object** is populated on log entries that were created to record the creation of some other object. For example, if an overdue event creates a customer contact, the related object contains information about the customer contact. Please note that if the object description is shown in blue, you can click on the object's description to drill down to the object.
  - **Related Process / Event** appears if the log entry was created by an overdue event. It references the unique identifier of the event.
  - **Log User** contains the user who caused the log entry to be created.
  - **Log Type** indicates how the log entry was created. The possible values are:
    - **User**. A user manually created this entry.

- **System.** This system created this entry.

## Maintaining Collection Cases

---

Use the Collection Case transaction to view and maintain pending or historic collection cases. Navigate using **Main Menu, Compliance, Collection Case**.

### Contents

- [Collection Case Query](#)
- [Collection Case Portal](#)
- [How To Perform Common Overdue Process Functions](#)

## Collection Case Query

Use the [query portal](#) to search for a collection case. Once a request is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Collection Case Portal

This portal appears when a collection case has been selected from the Collection Case Query portal.

The topics in this section describe the base-package zones that appear on this portal.

### Contents

- [Collection Case Actions](#)
- [Collection Case](#)
- [Collection Case Log](#)
- [Collection Case Overdue Processes](#)

## Collection Case Actions

This is a standard [actions zone](#).

If the collection case is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

In addition to possible next states, a collection case would typically include special action buttons. The action buttons would normally launch a BPA script that handles the additional processing required for the action; for instance, navigating to the appropriate page to create a pay plan.

If the collection case is in a final state, the state transition and action buttons are not displayed.

## Collection Case

The Collection Case zone contains display-only information about the selected Collection Case.

Please see the zone's help text for information about this zone's fields.

## Collection Case Log

This is a standard [log zone](#).

## Collection Case Overdue Processes

The Collection Case Overdue Processes zone contains display-only information about the overdue processes linked to the collection case.

## How To Perform Common Overdue Process Functions

The topics in this section describe how to perform common overdue process maintenance functions. Refer to [The Big Picture Of Overdue Processes](#) for high-level information about overdue processing.

### Contents

- [How To Create An Overdue Process](#)
- [How To Change Overdue Events](#)
- [How To Cancel An Overdue Process](#)

## How To Create An Overdue Process

99.9% of all overdue processes are created by the [Overdue Monitor](#) and require no human intervention. The other 0.1% are created by users on-line / real time. The following points describe how to create the 0.1%.

- Use **Control Central** to choose the account that requires a new overdue process.
- After the account is populated on Control Central, choose the **Overdue Process +** option on the account context menu to transfer to the overdue process transaction in add mode for the account.
- After the [Overdue Process - Main](#) page appears, specify the appropriate **Overdue Process Template**. The template is used to default the process's events (the template also controls numerous business rules, for example, when it will be canceled, how it will be canceled, etc). Refer to [How To Change Overdue Events](#) for a description of how you can override these events.
- Enter the overdue object(s) in the **Collecting On** grid. You must define at least one object.
- Save the overdue process.

## How To Change Overdue Events

When an overdue process is first created, it has one or more overdue events. The events are the activities that will be performed to persuade the taxpayer to pay the outstanding debt.

The number and type of events that are created when an overdue process is initiated are defined on the overdue process's overdue process template. The following points describe how to add / change / delete events on an overdue process if the defaulted events are not satisfactory.

- Use **Control Central** to choose the account with the overdue process whose events need to be changed.
- After the account is populated on Control Central, choose the **Overdue Process** option on the account context menu to transfer to the overdue process transaction in update mode for the account. Note that an account's **Active** overdue processes can be selected from the [Alert zone](#).



- To add a new event, transfer to the [Overdue Process - Events](#) tab and press the + button in the **Overdue Events** scroll. At this point, the event has not been added to the database; rather, it just exists in memory. Before you add the event to the database, you must specify the following information:
  - Choose an **Event Sequence** so that the new event will be positioned properly in respect of the other events.
  - Choose a **Status** of **Pending**.
  - Choose the desired **Overdue Event Type**.
  - If the activation of the new event is dependent on the successful completion of earlier events, turn on **Dep on Other Event** and then:
    - specify the sequences of the dependent events in **Event Dependencies** and
    - specify the how many days after the completion of the last dependent event that the new event should be triggered.
  - If the activation of the new event is NOT dependent on the successful completion of earlier events, turn off **Dep on Other Event** and use **Trigger Date** to define the date on which the event should be activated (i.e., completed).
- To delete an existing event, click on the event in the tree on [Overdue Process - Main](#). This will transfer you to the **Events** tab where you can click the - button to remove the event. At this point, the event has not been removed from the database; rather, it's been removed in memory.
- To change an existing event, click on the event in the tree on [Overdue Process - Main](#). This will transfer you to the **Events** tab where you can make the desired changes.
- After all desired changes have been made, save the overdue process.

## How To Cancel An Overdue Process

The system will cancel an overdue process when the [cancel criteria defined on the overdue process are satisfied](#). The following points describe how to manually cancel an overdue process.

- Use **Control Central** to choose the account with the overdue process to be cancelled.
- After the account is populated on Control Central, choose the **Overdue Process** option on the account context menu to transfer to the overdue process transaction in update mode for the account. A list of all overdue processes associated with the account appears. If only one overdue process exists, it is automatically selected for you. Note, an account's **Active** overdue processes can be selected from the [Alert zone](#).
- On the **Main** tab, click Cancel and answer any prompts.

## Collection Referral

The Collection Referrals page contains information about an account's debt that has been referred to a collection agency.

Refer to [Collection Agency Referrals](#) for more information about how the system creates and maintains collection agency referrals as part of overdue processing. In theory, you need only access this page if you need to override the automated processing.

Open **Compliance, Collection Agency Referral** to maintain this information.

### Description of Page

The **Agency Referrals** scroll contains one entry for every collection agency referral associated with the account. The following information is defined for each referral.

Use **Collection Agency** to define the agency to which the referral is being sent.

**Start Date** is the date on which the referral was initially created.

**Referral Status** defines if the referral is *Active* or *Closed*.

Use **Comments** to describe anything unusual about the referral.

The grid contains the history of interactions with the collection agency. Each time an account's debt is referred to a collection agency, the system creates a referral history record.

You can communicate changes about the referral by inserting a new row in the collection. For example,

- If you need to change the referral amount, insert a row and indicate the **Creation Date** and a **Referral History Reason** of *Change Referral*.
- If you need to cancel the referral, insert a row and indicate a **Referral History Reason** of *Referral Cancellation*.
- If the customer pays, insert a row and indicate a **Referral History Reason** of *Referral Paid*.
- If you need to add a new referral, insert a row and indicate a **Referral History Reason** of *Initial Referral*.

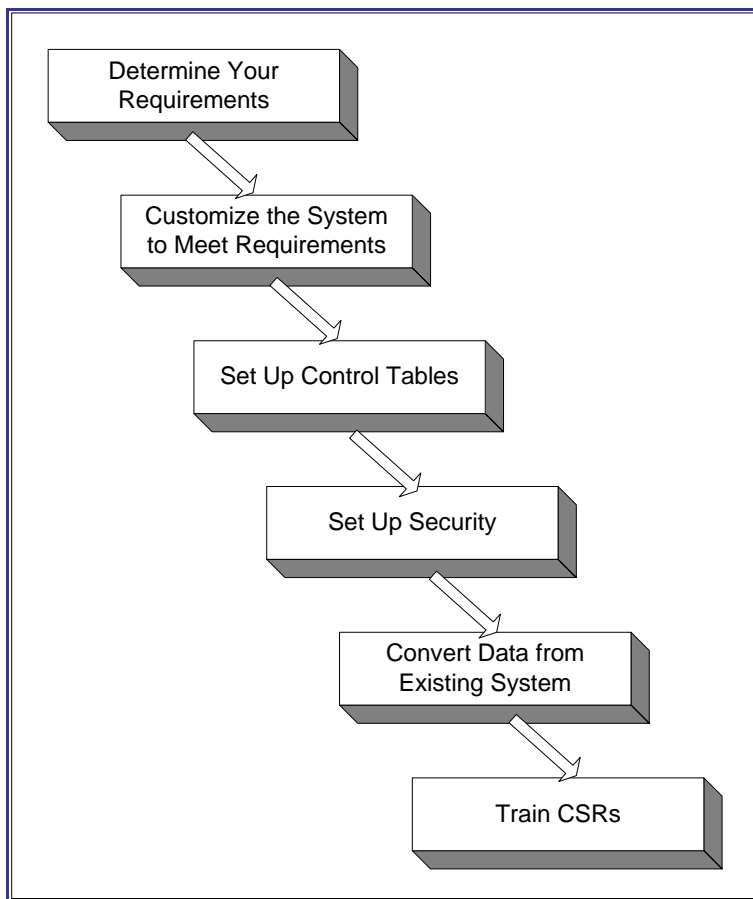
Collection agency referral records are interfaced to the respective collection agency using the batch process defined on the collection agency control record (refer to [Setting Up Collection Agencies](#)). The **Batch Control** process and the respective **Batch Number** in which the records were interfaced to the collection agency are displayed adjacent to the referral history reason.

The **System Generated** switch is on for those collection agency referrals created by other system events. Refer to [Collection Agency Referrals](#) for more information.

# Preparing To Implement

Getting ready for production takes a good deal of planning. You have probably already begun analyzing your requirements according to your business and organizational needs. You will need to review your current environment and think about what changes could be made now and in the future. And while you might have decided to simply transfer your current processing structure to Oracle Enterprise Taxation Management, you may also have discovered that Oracle Enterprise Taxation Management can provide new options.

Because the system is sophisticated and customizable, there are a number of steps involved in rolling out and using your new system.



The topics in this section describe the order in which the control tables should be set up.

## Contents

- [Control Table Setup Sequence](#)
- [Cross Reference To The Remaining Chapters](#)
- [Overdue Processing Setup Sequence](#)
- [Overpayment Processing Setup Sequence](#)
- [Penalty and Interest Setup Sequence](#)
- [Open-Item Accounting Table Setup Sequence](#)
- [Payment Event Distribution Table Setup Sequence](#)
- [Reports Setup Sequence](#)
- [XML Application Integration Setup Sequence](#)

[Case Management Setup Sequence](#)  
[Forms Processing Setup Sequence](#)  
[Zone Setup](#)  
[To Do Options Setup](#)  
[How To Copy An Algorithm From The Demonstration Database](#)

## Control Table Setup Sequence

To implement the system, you must set up your organization's business rules in "control tables". Setting up these tables is time-consuming because we allow you to tailor many aspects of the system to meet your organization's requirements. We strongly recommend that you take the time to document how you plan to set up all of these tables before you use the following roadmap to enter the control data. Time spent understanding the interrelationships between this data will reap the rewards of a clean system that meets your current and long term needs.

While we describe the transactions and options in more detail in other sections of this manual, use the following chart (and the remaining sections of this chapter) as your roadmap. Here we list the order in which you perform tasks and the pages you'll use to set up your system. The order is important because some information must exist before other information can be defined (i.e., many dependencies exist).

**Auto setup.** The Auto Setup column in the following table contains suggestions to save you time. It also indicates if a control table contains information when the system is installed.

**You don't have to set up every control table.** You need only set up those control tables that govern functions that are applicable to your organization.

Function	Menu	Auto Setup
<i>Global Context</i>		
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that populates global context values. The global context is used by various zones in the system to display relevant data. This algorithm is plugged-in on the <a href="#">installation record</a> .	You can run the <a href="#">CL_COPIN</a> DB process to copy many of the algorithms that support basic functionality from the demonstration database. Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
<i>Accounting Environment</i>		
Country & State	Admin Menu, Country	
Currency Codes	Admin Menu, Currency Code	<a href="#">USD</a> is automatically populated
Accounting Calendar	Admin Menu, Accounting Calendar	
GL Division	Admin Menu, General Ledger Division	
<i>Security Environment</i>		
Application Service	Admin Menu, Application Service	All base package transactions are automatically populated
Security Type	Admin Menu, Security Type	

Function	Menu	Auto Setup
User Group	Admin Menu, User Group Note, you won't be able to set up users at this point	One user group, <b>ALL-SERVICES</b> , is automatically set up. It references all other application services and a single user called <b>SYSUSER</b> .  Note: You may be able to <a href="#">import sample user groups from the demonstration database</a> . Also, you may be able to <a href="#">import user groups if your organization has already defined them using LDAP</a> .
Language	Admin Menu, Language	<b>ENG</b> is automatically populated
Display Profile	Admin Menu, Display Profile	Two display profiles are automatically set up: <b>NORTHAM</b> displays currencies and dates in a classic American format; <b>EURO</b> displays information in a classic European format
Data Access Role	Admin Menu, Data Access Role	
Access Group	Admin Menu, Access Group	
User	Admin Menu, User	<b>SYSUSER</b> is automatically set up. Note, you may be able to <a href="#">import your users if your organization has already defined them using LDAP</a> .
Return to User Group	You must return to your user groups and define all of their users	
<b>Account Type Environment</b>		
Account Type	Admin Menu, Account Type. At this point, you'll only be able to set up your account type codes. You will return to these account types throughout the setup process to populate additional information.	
<b>Financial Transaction Environment</b>		
Work Calendar	Admin Menu, Work Calendar	
Division	Admin Menu, Division	
Revenue Class	Admin Menu Revenue Class	
Algorithm	Admin Menu, Algorithm. You will need to set up the algorithm that constructs a distribution code's corresponding GL account when it is interfaced to the general ledger	
Distribution Code	Admin Menu, Distribution Code	

Function	Menu	Auto Setup
Bank & Bank Accounts	Admin Menu, Bank	
Billable Charge Template	Admin Menu, Billable Charge Template. Note, if you want the system to default rate quantities onto billable charges created using this template, you must set up the appropriate unit of measure code, time-of-use code and/or rate quantity identifier.	
Billable Charge Upload Line Type	Admin Menu, Billable Charge Line Type	
Algorithm	Admin Menu, Algorithm. You will need to set up several algorithms. These algorithms: 1) retrieve a bill segment's consumption, 2) calculate a bill segment's bill lines, 3) construct a bill segment's financial transaction, 4) cancel previously estimated bill segments	Rather than setting these up manually, you can run the <a href="#"><i>CL_COPBI</i></a> DB process to copy many of these algorithms from the demonstration database. Please review the parameter values on these algorithms after they are copied. Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
Bill Segment Type	Admin Menu, Bill Segment Type	
Algorithm	Admin Menu, Algorithm. You will need to set up the algorithm that constructs a payment segment's financial transaction	Rather than setting these up manually, you can run the <a href="#"><i>CL_COPPY</i></a> DB process to copy many of these algorithms from the demonstration database. Please review the parameter values on these algorithms after they are copied. Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
Payment Segment Type	Admin Menu, Payment Segment Type	
Algorithm	Admin Menu, Algorithm. You will need to set up the algorithm that constructs an adjustment's financial transaction	Rather than setting these up manually, you can run the <a href="#"><i>CL_COPAD</i></a> DB process to copy many of these algorithms from the demonstration database. Please review the parameter values on these algorithms after they are copied. Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
Algorithm	Admin Menu, Algorithm. Several plug-in spots are available to perform additional logic when processing adjustments. For example, if you have the system calculate adjustments, you must set up an	

Function	Menu	Auto Setup
	adjustment generation algorithm. Refer to <a href="#">Adjustment Type</a> for other available plug-in spots that may be used by your implementation.	
Algorithm	Admin Menu, Algorithm. You may want to set up an algorithm that formats the Adjustment information that is displayed throughout the system for a specific Adjustment Type. This algorithm is plugged-in on the <a href="#">Adjustment Type</a> .	
Algorithm	Admin Menu, Algorithm. You may want to set up an algorithm that formats the Adjustment information that is displayed throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	
Debt Category	Admin Menu, Debt Category	
Debt Category Priority	Admin Menu, Debt Category Priority	
Adjustment Type	Admin Menu, Adjustment Type	
Adjustment Type Profile	Admin Menu, Adjustment Type Profile	
Approval Profile	Admin Menu, Approval Profile. Note, an approval profile references a To Do type and one or more To Do Roles; these must be set up before you can set up the approval profile. After the approval profile(s) are set up, they must be referenced on the adjustment types that they govern.	
Cancel Reason – Bill	Admin Menu, Bill Cancel Reason	
Cancel Reason – Payment	Admin Menu, Payment Cancel Reason	
Cancel Reason – Adjustment	Admin Menu, Adjustment Cancel Reason	
Tender Type	Admin Menu, Tender Type	
Tender Source	Admin Menu, Tender Source	
A/P Request Type	Admin Menu, A/P Request Type	

Function	Menu	Auto Setup
Installation	Admin Menu, Installation Options - Framework and Admin Menu, Installation Options. Many fields on the installation record impact the financial transaction environment. Refer to the description of the <a href="#">Billing</a> and <a href="#">Financial Transaction</a> tabs and the <a href="#">Messages</a> tab in the Framework page for more information.	
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that distributes payments.	If you ran the <a href="#">CL_COPY</a> DB process described above, this algorithm will have been set up for you.
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that handles overpayment situations.	
Algorithm	Admin Menu, Algorithm. You may need to set up an algorithm if specific taxpayers can have individual bill due dates.	
Algorithm	Admin Menu, Algorithm. You may need to set up an algorithm if you want the system to levy a non-sufficient funds charge if a payment is canceled due to non-sufficient funds.	
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that formats the bill information that is displayed throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	You can run the <a href="#">CL_COPIN</a> DB process to copy many of the algorithms that format basic information from the demonstration database. Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that formats the payment information that is displayed throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPIN</a> DB process described above, this algorithm will have been set up for you
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that defaults the amount when a payment is manually added. This algorithm also calculates the amount of an automatic payment for a bill for an account with an active auto pay option. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPIN</a> DB process described above, this algorithm will have been set up for you



Function	Menu	Auto Setup
Algorithm	Admin Menu, Algorithm. Refer to <a href="#">Account Type</a> for other available plug-in spots that may be used by your implementation to perform additional logic when processing payments and bills.	
Return to Account Type	Admin Menu, Account Type. You will need to plug-in the algorithms defined above on your account types.	
<b>Customer Environment</b>		
Account Management Group	Admin Menu, Account Management Group. Note: You will probably have to set up To Do Type and To Do Roles before you can set up account management groups. Refer to <a href="#">Assigning A To Do Role</a> for more information on how account management groups may be used to define an entry's role.	
Account Relationship	Admin Menu, Account Relationship Type	
Alert Type	Admin Menu, Alert Type	
Bill Message	Admin Menu, Bill Message	
Algorithm	Admin Menu, Algorithm. If you have software capable of reconstructing an image of a bill in a PDF (for the purpose of online display), you will need to create an algorithm that formats the extract records that are sent to your bill image software.	Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
Bill Route Type	Admin Menu, Bill Route Type	
Contract Quantity Type	Admin Menu, Contract Quantity Type	
Algorithm	Admin Menu, Algorithm. If you have software capable of reconstructing an image of a letter in a PDF (for the purpose of online display), you will need to create an algorithm that formats the extract records that are sent to your letter image software.	If you ran the <a href="#">CL_COPD1</a> DB process described above, these algorithms will have been set up for you.
Letter Template	Admin Menu, Letter Template	
Customer Contact Class	Admin Menu, Customer Contact Class	
Customer Contact Type	Admin Menu, Customer Contact Type	

Function	Menu	Auto Setup
Algorithm	Admin Menu, Algorithm. You may need to set up the algorithms that determine if person ID's are in a predefined format.	
Identifier Type	Admin Menu, Identifier Type	
Industry Codes	Admin Menu, Industry Code	
Tax Exempt Type	Admin Menu, Tax Exempt Type	
Algorithm	Admin Menu, Algorithm. You may need to set up the algorithms that determine if phone numbers are in a predefined format.	
Phone Type	Admin Menu, Phone Type	
Person Relationship Type	Admin Menu, Person Relationship Type.	
Person Type	Admin Menu, Person Type	
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm that formats the person information that is displayed throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPIN</a> DB process described above, this algorithm will have been set up for you
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm to validate a person's name. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPIN</a> DB process described above, this algorithm will have been set up for you
Algorithm	Admin Menu, Algorithm. You can override the system's standard account information string by setting up an algorithm that produces this string of information. This algorithm is plugged-in on the <a href="#">installation record</a> .	
Algorithm	Admin Menu, Algorithm. If you have software capable of reconstructing an image of a letter in a PDF for the purpose of online display, you will need to create an algorithm that renders this PDF. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPD1</a> DB process described above, this algorithm will have been set up for you
Algorithm	Admin Menu, Algorithm. If you have software capable of reconstructing an image of a bill in a PDF for the purpose of online display, you will need to create an algorithm that renders this PDF. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPD1</a> DB process described above, this algorithm will have been set up for you

Function	Menu	Auto Setup
Installation	Admin Menu, Installation Options. Many fields on the installation record impact the Customer Environment. Refer to the description of the <a href="#">Main</a> , <a href="#">Person</a> , and <a href="#">Account</a> tabs for more information.	
<b><i>Automatic Payment (EFT) Environment</i></b>		
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm to create automatic payments. This algorithm is plugged-in on the <a href="#">installation record</a> .	You can run the <a href="#">CL_COPAP</a> DB process to copy this algorithm (and other autopay-oriented algorithms) from the demonstration database. Refer to <a href="#">How To Copy An Algorithm From The Demo Database</a> for more information.
Tender Source	Admin Menu, Tender Source Note: Earlier, you created tender sources for the remittance processor and your cash drawers. At this point, you'll need to add at least one tender source for automatic payments. Why? Because automatic payments get linked to a tender control (which, in turn, gets linked to a tender source) when they are interfaced out of the system.	
Algorithm	Admin Menu, Algorithm. You will need to set up the appropriate automatic payment date calculation algorithm to populate the extract, GL interface and payment dates on automatic payments.	If you ran the <a href="#">CL_COPAP</a> DB process described above, this algorithm will have been set up for you
Auto Pay Route Type	Admin Menu, Auto Pay Route Type	
Tender Type	Admin Menu, Tender Type Note: Earlier, you created tender types for things like cash, checks, etc. At this point, you'll need to add a tender type for each type of automatic payments (e.g., direct debt, credit card, etc.).	
Work Calendar	Admin Menu, Work Calendar. You need only set up additional work calendars if the auto pay sources (i.e., the financial institutions) have different working days than does your organization	

Function	Menu	Auto Setup
Algorithm	Admin Menu, Algorithm. If you need to validate the taxpayer's bank account or credit card number, you will need to set up the appropriate validation algorithms.	If you ran the <a href="#">CL_COPAP</a> DB process described above, this algorithm will have been set up for you
Auto Pay Source Type	Admin Menu, Auto Pay Source Type	
Algorithm	Admin Menu, Algorithm. You may need to set up an algorithm if your taxpayers can define a maximum withdrawal limit on their autopay options.	If you ran the <a href="#">CL_COPAP</a> DB process described above, this algorithm will have been set up for you
Return to Account Type	Admin Menu, Account Type. You should plug-in the Autopay Over Limit Algorithm in each appropriate Account Type.	
<b>Characteristics</b>		
Algorithm	Admin Menu, Algorithm. If you have ad hoc characteristic types, you may need to set up the algorithms that control how they are validated	
Foreign Key Reference	Admin Menu, FK Reference. If you have foreign key characteristic types, you may need to set up foreign key references to control how the user selects the characteristic values (and how the foreign key values are validated).	All base package FK references are automatically populated
Characteristic Type & Values	Admin Menu, Characteristic Type	
<b>Asset Environment</b>		
Manufacturer / Model	Admin Menu, Manufacturer	
Unit of Measure	Admin Menu, Unit of Measure	
Time of Use	Admin Menu, Time of Use	
Retirement Reason	Admin Menu, Retire Reason	
Asset Type	Admin Menu, Asset Type	
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm to format the standard item info that appears throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPIN</a> DB process described above, this algorithm will have been set up for you
<b>Location Environment</b>		
Location Type	Admin Menu, Location Type	

Function	Menu	Auto Setup
Algorithm	Admin Menu, Algorithm. You will need to set up an algorithm to format the standard location info that appears throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	If you ran the <a href="#">CL_COPIN</a> DB process described above, this algorithm will have been set up for you
Algorithm	Admin Menu, Algorithm. You may need to set up the algorithms that determine if geographic ID's are in a predefined format.	
Geographic Type	Admin Menu, Geographic Type	
<b>Rate Environment</b>		
Frequency	Admin Menu, Frequency	
Rate Quantity Identifier	Admin Menu, Rate Quantity Identifier	
Algorithm Type	Admin Menu, Algorithm Type. If you create new Rate Quantity Rules you must set up an algorithm type for each such rule (the algorithm type defines the types of parameters that are passed to the RQ rule).	All base package algorithm types are automatically populated
Rate Quantity Rule	Admin Menu, Rate Quantity Rule	
Rate Factor	Main Menu, Rates, Rate Factor	
Rate	Main Menu, Rates, Rate Schedule	
Rate Version	Main Menu, Rates, Rate Version	
Algorithm	Admin Menu, Algorithm. If you use algorithms to dynamically change step boundaries, calculate prices, or implement rate component eligibility rules, you must set up these algorithms.	
Rate Component	Main Menu, Rates, Rate Component	
Rate Factor Value	Main Menu, Rates, Rate Factor Values	
Asset Type RQ Estimate	Admin Menu, Asset Type RQ Estimate	
<b>Obligation Configuration</b>		
Filing Calendar	Admin Menu, Filing Calendar	
Tax Type	Admin Menu, Tax Type	

Function	Menu	Auto Setup
Algorithm	Admin Menu, Algorithm. You will need to set up the algorithms that determine: <ul style="list-style-type: none"> <li>• Special criteria to be tested before an obligation is severed.</li> <li>• Special processing that should take place prior to the completion of a bill that references obligations of a given type.</li> <li>• Special processing that should take place during completion of a bill that references obligations of a given type.</li> <li>• Special processing that should take place when obligations of a given type are created.</li> <li>• Special processing that should take place when a financial transaction is frozen for obligations of a given type.</li> </ul>	
Algorithm	Admin Menu, Algorithm. You may want to set up an algorithm that formats the obligation information that is displayed throughout the system. This algorithm is plugged-in on the <a href="#">installation record</a> .	
Algorithm	Admin Menu, Algorithm. You may want to set up an algorithm that formats the obligation information that is displayed throughout the system for a specific obligation type. This algorithm is plugged-in on the <a href="#">Obligation Type</a> .	
Bill Cycle, Bill Cycle Schedule	Admin Menu, Bill Cycle	
Bill Period, Bill Period Schedule	Admin Menu, Bill Period	
Obligation Type	Admin Menu, Obligation Type	
Obligation Type Start Options	Admin Menu, Obligation Type Start Option	
<b>Wrap Up</b>		
Algorithm	Admin Menu, Algorithm. You will need to set up the algorithms that determine:	

Function	Menu	Auto Setup
	<ul style="list-style-type: none"> <li>Special alerts on Control Central (assuming you have special alerts)</li> </ul>	
Installation Options	Admin Menu, Installation Options - Framework and Admin Menu, Installation Options. At this point, it's a good idea to double-check everything on the installation record.	
Postal Default	Admin Menu, Postal Code Default	

If you have cash drawers you will also need to set up the following information:

- Create a person / account to which you will link your over / under obligation. Refer [How To Get An Unbalanced Tender Control In Balance \(Fixing Over/Under\)](#) for more information.
- Create an obligation to which your over/under payments will be linked. This obligation will reference your over / under obligation type. Refer to [Over / Under Cash Drawer Obligations](#) for more information.

If you upload payments from an external source (e.g., a remittance processor or lock box), you must set up the following information:

- Create a person and account to which the system will link payments with invalid account. Refer to [Phase 3 – Create Payment Events, Tenders, Payments and Payment Segments](#) for information about the process that books invalid payments to this account. Refer to [How To Transfer A Payment From One Account To Another](#) for how a user transfers the payment from the invalid account to the correct account (once known).
- Create an obligation for this account. This obligation will reference your payment suspense obligation type. The system needs this obligation so that it can distribute the invalid account's payment (and this is necessary so that cash will reflect the payment). Refer to [Payment Upload Error Obligations](#) for more information.
- Update the tender source associated with the respective source of payments to indicate the obligation created in the previous step should be used for payments with invalid accounts. Refer to [Setting Up Tender Sources](#) for more information.
- Because the payment upload process simply books payments that reference invalid accounts to the account associated with the suspense obligation on the payment's tender source, this account should belong to an account type with the appropriate payment distribution algorithms. This may entail creating a new account type that will only be used on suspense accounts.

The remaining sections describe additional control tables that must be set up for specific functional areas.

## Cross Reference To The Remaining Chapters

The table in the previous section describes the order in which you should enter your control tables. These tables are described at length in the following chapters.

- Refer to [Defining General Options Addendum](#) and [Defining General Framework Options](#) for a discussion of the control tables associated with general functionality (e.g., country codes, state codes, etc.).
- Refer to [Defining Financial Transaction Options](#) for a discussion of the tables affecting your financial transactions (e.g., bill segment types, payment segment types, etc.).
- Refer to [Defining Taxpayer Options](#) for a discussion of the control tables affecting persons, accounts and obligations.
- Refer to [Rates](#) for a discussion of the control tables affecting your rates.
- Refer to [Defining Obligation Types](#) for a discussion of the control tables affecting your obligation types.
- Refer to [Defining Background Process](#) for a discussion of the control tables affecting your background processes.
- Refer to [Defining Algorithms](#) for a discussion of the control tables affecting the algorithms referenced on many control tables.

## Overdue Processing Setup Sequence

---

Overdue processing tables need to be set up for collection activity. Refer to [Overdue Processing - Setup Tasks](#) for a description of tables that must be set up to enable this functionality.

## Overpayment Processing Setup Sequence

---

Overpayment processing tables need to be set up for overpayment activity. Refer to [Setting Up Overpayment Options](#) for a description of tables that must be set up to enable this functionality.

## Penalty and Interest Setup Sequence

---

Penalty and interest related configuration tables need to be set up for penalty and interest calculations. Refer to [Setting Up Penalty and Interest Options](#) for a description of tables that must be set up to enable this functionality.

## Open-Item Accounting Table Setup Sequence

---

Open-item accounting tables need only be set up if your organization practices [Open Item Accounting](#). Refer to [Setting Up The System To Enable Open Item Accounting](#) for a description of the tables that must be set up to enable this functionality.



## Payment Event Distribution Table Setup Sequence

---

Payment event distribution tables need only be set up if your organization opted to use the distribution rules method to create payment events. Refer to [Setting Up The System To Use Distribution Rules](#) for a description of the tables that must be set up to enable this functionality.

## Reports Setup Sequence

---

In order to use the reporting tool, you will need to set up reporting options. Refer to [Configuring The System To Enable Reports](#) for more information.

**Importing sample reports.** Refer to [How To Copy A Report From The Demonstration Database](#) if you want to import sample report metadata from the demonstration database.

## XML Application Integration Setup Sequence

---

In order to use the XAI tool for sending information between third parties, you will need to set up XAI control tables. Refer to [XML Application Integration](#) for more information.

## Case Management Setup Sequence

---

Case management options need only be set up if your organization uses cases to manage issues. Refer to [Setting Up Case Types](#) for more information.

## Forms Processing Setup Sequence

---

Forms processing tables need to be set up for processing registration forms, tax forms and form batch headers (if applicable). Refer to [Defining Forms Processing Options](#) for more information.

## Zone Setup

---

Most zones are delivered with the base-package and do not require any configuration. However, some zones are only available if configured by your implementation. Refer to [Configuring Zones](#) for more information.

## To Do Options Setup

---

Refer to [Setting Up To Do Options](#) for more information on how to configure the system to match your organization's To Do management needs.

## How To Copy An Algorithm From The Demonstration Database

---

**Warning!** If you are not familiar with the concepts described in the [ConfigLab](#) chapter, this section will be difficult to understand. Specifically, you need to understand how a *Compare* database process is used to copy objects between two databases. Please take the time to familiarize yourself with this concept before attempting to copy an algorithm from the demonstration database.

The demonstration database contains many sample algorithms. The topics in this section describe how to copy a subset of the demonstration algorithms to your implementation's database.

### Contents

- [If You Work In A Non-English Language](#)
- [One Time Only - Set Up A DB Process To Copy Algorithms](#)
- [Run The Copy Algorithms DB Process](#)

## If You Work In A Non-English Language

The demonstration database is installed in English only. If you work in a non-English language, you must execute the [NEWLANG](#) background process on the demonstration database before using it as a *Compare Source* supporting environment. If you work in a supported language, you should apply the language package to the demonstration database as well.

If you don't execute NEWLANG on the demonstration database, any objects copied from the demonstration database will not have language rows for the language in which you work and therefore you won't be able to see the information in the target environment.

## One Time Only - Set Up A DB Process To Copy Algorithms

You need a "copy algorithm" [database process](#) (DB process) set up in the target database (i.e., your implementation's database). This DB process has a single instruction that references the algorithm [maintenance object](#) (MO). This instruction should have a table rule with an override condition that selects the algorithms in question. For example, the override condition, `#CI_ALG.ALG_CD IN ('ADJT-AC', 'ADJT-AD', 'ADJT-CA', 'ADJT-GL', 'ADJT-NM', 'ADJT-RA', 'ADJT-TC')`, is used on the DB process that copies specific adjustment type algorithms.

The demonstration database contains several such a DB processes, for example:

- **CI\_COPAD** copies algorithms that control how adjustment financial transactions are created.
- **CI\_COPAP** copies algorithms that control how automatic payments are created.
- **CI\_COPBI** copies algorithms that control how bill segments and their financial transactions are created.
- **CI\_COPIN** copies various installation algorithms that control validation, formatting and other general functionality.
- **CI\_COPPY** copies algorithms that control payment processing.
- ...

In order to copy algorithms from the demonstration database, you must first copy these DB processes from the demonstration database.

**Warning!** The remainder of this section is confusing as it describes a DB process that copies other DB processes. Fortunately, you will only have to do the following once. This is because you only have to copy these DB processes once.

You can copy these DB processes from the demonstration database by submitting the **CL-COPDB** background process in your target database. When you submit this process, you must supply it with an [environment reference](#) that points to the demonstration database. If you don't have an environment reference set up in your target database that references the demonstration database, you must have your technical staff execute a registration script that sets up this environment reference. Refer to [Registering ConfigLab Environments](#) for more information.

**CL-COPDB** is initially delivered ready to copy every DB process that is prefixed with **CL** from the source database (there are numerous sample DB processes in the demonstration database and this process copies them all). If you only want to copy only the DB processes described above, add a table rule to the primary instruction of the **CL-COPDB** database process to only copy the desired processes. If you don't add this rule, all DB processes in the demonstration will be copied to your target database (and this might be exactly what you want).

When the **CL-COPDB** process runs, it highlights differences between the DB processes in your source database and the target database. The first time you run this process, it creates a root object in the target database for each DB process that will be added to your target database. You can use the [Difference Query](#) to review these root objects and **approve** or **reject** them.

**Automatic approval.** When you submit **CL-COPDB**, you can indicate that all root objects should be marked as **approved** (thus saving yourself the step of manually approving them using [Difference Query](#)).

After you've approved the root object(s), submit the **CL-APPCH** batch process to change your target database. You must supply the **CL-APPCH** process with two parameters:

- The DB Process used to create the root objects (**CL-COPDB**)
- The environment reference that identifies the source database (i.e., the demonstration database)

## Run The Copy Algorithms DB Process

After you have populated the “copy algorithms” DB processes in your target database, you can override their [table rules](#) to edit the list of algorithms that will be copied. You need only do this if you don't need all of the algorithms that are defined in these DB processes (but it never hurts to have too many algorithms as they won't be used unless you plug them in on the appropriate object).

At this point, you're ready to submit the background process identified on your “copy algorithm” DB processes. These background processes highlight the differences between the algorithms in the demonstration database and the target database (the target database is the environment in which you submit the background process).

**The background process you submit is typically named the same as the DB process that contains the rules.** If you used the **CL-COPDB** background process to transfer the “copy algorithm” DB processes from the demo database, it will have also set up these batch controls and linked to each the appropriate “copy algorithms” DB process. These batch controls have the same name as their related DB process (this is just a naming convention, it’s not a rule). This means, for example, that you’d submit a batch control called **CI\_COPAD** in order to execute the **CI\_COPAD** DB process.

When you submit one of the DB processes defined above, you must supply it with an [environment reference](#) that points to the source database (i.e., the demonstration database).

When the process runs, it simply highlights differences between the algorithms in the source database and the target database. It creates a root object in the target database for every algorithm that is not the same in the two environments (actually, it only concerns itself with algorithm that match the criteria on the DB process's table rule described above). You can use the [Difference Query](#) to review these root objects and **approve** or **reject** them.

**Auto approval.** When you submit the process, you can indicate that all root objects should be marked as **approved** (thus saving yourself the step of manually approving them).

After you’ve approved the root object(s) associated with the algorithms that you want copied, submit the **CL-APPCH** batch process to cause your target database to be changed. You must supply the **CL-APPCH** process with two parameters:

- The DB process of the “copy algorithms” DB process (e.g., **CI\_COPAD**)
- The environment reference that identifies the source database (i.e., the demonstration database)

# Defining General Options Addendum

This section describes control tables that are used throughout Oracle Enterprise Taxation Management.

## Contents

- [Defining Installation Options](#)
- [Defining Taxpayer Languages](#)
- [Defining Accounting Calendars](#)
- [Defining General Ledger Divisions](#)
- [Defining Banks & Bank Accounts](#)
- [To Do Lists Addendum](#)

## Defining Installation Options

The topics in this section describe the various installation options that control various aspects of the system that are specific to the Oracle Enterprise Taxation Management product.

Refer to [Installation Options - Framework](#) for options that are common to products on the same framework.

## Contents

- [Installation Options - Main](#)
- [Installation Options - Person](#)
- [Installation Options - Account](#)
- [Installation Options - Billing](#)
- [Installation Options - Collections](#)
- [Installation Options - Financial Transaction](#)
- [Installation Options - Algorithms](#)

## Installation Options - Main

Select **Admin Menu, Installation Options** and use the **Main** tab to define system-wide installation options.

### Description of Page

Use **Quick Add Tender Type** to define the tender type defaulted on payments added using the [Payment QuickAdd](#) transaction.

Use **Starting Balance Tender Type** to define the tender type of the starting balance recorded on your tender controls (this will almost always be the tender type associated with "cash"). This value is used during tender control balancing as a separate balance is required for each tender type in order to balance a tender control. Refer to [The Lifecycle Of A Tender Control](#) for more information.

For more information, refer to [Setting Up Tender Types](#).

Use the **Location Geo Type** to indicate whether at least one geographic identifier (e.g., GPS coordinate) is **Required** or **Optional** on a location. Refer to [Defining Geographic Types](#) for more information.

The **Alternate Representation** flag should be set to **None** unless your organization uses multiple character sets for a person's main name and / or a location's address. Alternate representations are typically only used in countries where multiple character sets are used. For example,

- In Hong Kong, a person's name may be written in both Chinese characters and in English.
- In Japan, a person's name may be written in both Kanji and Katakana.

In both of the above situations, users need to be able to use both representations to find a taxpayer or a location.

**Alerts that should appear adjacent to a person's name or address.** If your organization doesn't use multiple character sets, you might want to consider using this functionality to implement critical person or location alerts. For example, if you have a taxpayer who's supported by a specific account representative, you could enter the account rep's name as the person's alternate name. If you do this, the account rep's name would appear in parenthesis following the taxpayer's name. In addition, you can search for the taxpayers supported by the account rep on Control Central by entering the account rep's name.

If your organization uses alternate representations of person name or address, set this flag to one of the following values:

- Use a value of **Address** if you only use alternate representations for location addresses.
- Use a value of **Name** if you only use alternate representations for a person's primary names.
- Use a value of **Name & Address** if you use alternate representations for both location addresses and person names.

The following points describe the ramifications of this flag in the system:

- If you support alternate representations of a person's primary name,
  - The name grid on [Person - Main](#) allows you to specify an **Alternate** name for the person.
  - If you use the base package [name formatting algorithm](#), a person's name will be shown throughout most of the system in the format AAA (BBB), where AAA is the person's primary name and BBB is the person's alternate name. Note, this format does not apply to names that appear in search results (i.e., the alternate name is not concatenated to the main name in search results; however you can search for information using the alternate name).
  - Most of the system's person name-oriented searches will allow users to use both a person's primary and alternate names to search for information.
- If you support alternate representations of a location's address,
  - A new tab is available on the [Location](#) page that allows a user to define an alternate address for a location.
  - If you use the base package [premise formatting algorithm](#), a location's address will be shown throughout most of the system in the format AAA (BBB), where AAA is the location's primary address and BBB is the location's alternate address.

- Most of the system's location-oriented searches will allow users to use both a location's primary and alternate addresses to search for information.

Set the **CTI Integration** flag to **Yes** if your organization integrates with an external computer telephony integration (CTI) system that supports a "get next caller in the queue" function. If this flag is set to **Yes**, then [Next Call](#) button will appear in the action toolbar allowing customer service representatives to request the next taxpayer waiting in the queue to speak to a CSR.

**Warning!** In order to improve response times, installation options are cached the first time they are used after a web server is started. If you change this field's option and you don't want to wait for the cache to rebuild, you must clear the cached information so it will be immediately rebuilt using current information. Refer to [Caching Overview for information](#) on how to clear the system login cache (this is the cache in which installation options are stored).

## Installation Options - Person

Select **Admin Menu, Installation Options** and use the **Person** tab to define person-specific installation options.

### Description of Page

Use the **Person ID Usage** to indicate whether or not at least one form of identification is **Required** or **Optional** when a new person is added.

Each form of identification has an identifier type. For persons that are humans (as defined by the [person type](#)'s person/business identifier), the [Person](#) page defaults the identifier type defined in **Identifier Type (Person)**. For persons that are businesses (as defined by the [person type](#)'s person/business identifier), the [Person](#) page defaults the identifier type defined in **Identifier Type (Business)**.

## Installation Options - Account

Select **Admin Menu, Installation Options** and use the **Account** tab to define account-specific installation options.

### Description of Page

When a new account is added on the [Account](#) page, the system requires it have an account type. If the main taxpayer linked to the account is a human (as defined by the taxpayer's person type), the system defaults the account type defined in **Account Type (Person)**. For persons that are businesses (as defined by the person type), the system defaults the account type defined in **Account Type (Business)**. For more information, refer to [Setting Up Account Types](#).

In addition to requiring an account type when a new taxpayer is added, the system also requires a "main taxpayer" (i.e., a reference to a person who is identified as the main taxpayer for the account). Enter the default **Account Relationship Type Code** to be used to define the main taxpayer relationship. For more information, refer to [Setting Up Account Relationship Codes](#).

Enter the default **Bill Route Type** to be used to define how bills should be routed to a taxpayer. For more information, refer to [Setting Up Bill Route Types](#).

## Installation Options - Billing

Select **Admin Menu, Installation Options** and use the **Billing** tab to define billing-specific installation options.

### Description of Page

The **Bill Segment Freeze Option** controls when an obligation's balance and the general ledger are affected by bill segments and certain types of adjustments. Refer to [Preventing Obligation Balances And The GL From Being Impacted Until Bill Completion](#) to understand the significance of this option.

The **Accounting Date Freeze Option** controls how the accounting date defined on financial transactions is populated. Refer to [Forcing The Freeze Date To Be Used As The Accounting Date](#) to understand the significance of this option.

Define the **Minimum Amount for Final Bill**. If a final bill is less than this amount, the bill is still produced; it's just not printed.

Typically, the system sets a bill's Bill Date equal to the date on which it is completed. If you want to be able to specify a bill's Bill Date when you complete a bill, turn on **User Can Override Bill Date**. You would only want to override the bill date if you are setting up sample bills from historical period whose bill date needs to reflect the respective historical period.

## Installation Options - Collections

Select **Admin Menu, Installation Options** and use the **Collections** tab to define collections-specific installation options.

### Description of Page

Enter what you consider to be an excellent compliance rating in **Beginning Compliance Rating**. Overdue events can cause an account's compliance rating to decrease. When an account's compliance rating falls below a certain level, different overdue processes may ensue.

Use **Compliance Rating Threshold** to define when an account's compliance rating becomes risky. When an account's compliance rating falls beneath the Compliance Rating Threshold, the system will show an alert in Control Central highlighting such.

## Installation Options - Financial Transaction

Select **Admin Menu, Installation Options** and use the **Financial Transaction** tab to define financial transaction installation options.

### Description of Page

Use **G/L Batch Code** to define the batch process that is used to interface your financial transactions to your general ledger. The process is snapped on FT download records by the GLS background process.

Use **A/P Batch Code** to define the batch process that is used to interface your check requests (initiated with adjustments with an adjustment type that reference an A/P request type) to your accounts payable system.

Use the **Fund Accounting** to indicate if [fund accounting](#) link is **Practiced** or **Not Practiced** at your organization. By default, the installation indicates that it is **Practiced**.



## Installation Options - Algorithms

The following table describes each **System Event**.

System Event	Optional / Required	Description
<i>Account Information</i>	Optional	<p>We use the term "Account information" to describe the basic information that appears throughout the system to describe an account. The data that appears in "account information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the system default "Account information".</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Address Information</i>	Optional	<p>This algorithm allows your implementation to define a common format for displaying an address. It receives address constituents and returns a formatted address string. This plug-in spot may be invoked by other algorithms using the business service <i>C1-FormatAddressInfo</i>.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Adjustment Information</i>	Optional	<p>We use the term "Adjustment information" to describe the basic information that appears throughout the system to describe an adjustment. The data that appears in "Adjustment information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the system default "Adjustment information".</p> <p>Note: This algorithm may be further overridden by an "Adjustment information" plug-in on the Adjustment Type. Refer to <a href="#">Adjustment Type</a> for how algorithms of this type are used.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Automatic Payment Creation</i>	Required if you allow taxpayers to <a href="#">pay automatically</a>	<p>This algorithm is executed to create automatic payments whenever the system creates automatic payments. Refer to <a href="#">How And When Are Automatic Payments Created</a> for the details.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Bill Information</i>	Required	<p>We use the term "Bill information" to describe the basic information that appears throughout the system to describe a bill. The data that appears in "bill information" is constructed using this algorithm.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Case Information</i>	Optional	<p>We use the term "Case information" to describe the basic information that appears throughout the system to describe a case. The data that appears in "case information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the system default "Case information".</p> <p>Note: This algorithm may be further overridden by a "Case information" plug-in on the Case Type. Refer to <a href="#">Case Type</a> for how algorithms of this type are used.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Collection Agency</i>	Optional	<p>We use the term "Collection Agency Referral information" to describe the basic information that appears throughout the system to describe a collection agency</p>

<b><i>Referral Information</i></b>		<p>referral.</p> <p>Plug an algorithm into this spot to override the system default "collection agency referral information".</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Compliance Rating "Created By" Information</i></b>	Required	<p>The data that appears in the compliance rating "created by" information is constructed using this algorithm.</p> <p>Refer to <a href="#">Account - Collections</a> for more information about the compliance rating.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Compliance Rating History Information</i></b>	Optional	<p>We use the term Compliance Rating History information to describe the basic information that appears throughout the system to describe a <a href="#">compliance rating history</a> entry.</p> <p>Plug an algorithm into this spot to override the system default "compliance rating history information".</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Control Central Alert</i></b>	Optional	<p>There are two types of alerts that appear in the <a href="#">Alert Zone</a> and on <a href="#">Payment Event – Main</a>: 1) hard-coded system alerts and 2) alerts constructed by plug-in algorithms. You cannot change the hard-coded alerts (see the <a href="#">Alert Zone</a> for the complete list). However, by plugging in this type of algorithm you can introduce additional alerts.</p> <p>An error displays if more than 60 alerts are generated for an account by plug-in algorithms.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Determine Open Item Bill Amounts</i></b>	Required if you use overdue functionality to <a href="#">collect on bills</a>	<p>This algorithm is responsible for determining the unpaid amount of an open-item bill. It can also be used to return the unpaid amount for a specific Obligation on a bill.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Financial Transaction Info</i></b>	Optional	<p>We use the term "financial transaction information" to describe the basic information that appears throughout the system to describe a financial transaction. The data that appears in "financial transaction info" is constructed using this algorithm.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Location Information</i></b>	Required	<p>We use the term "location info" to describe the basic information that appears throughout the system to describe a location. The data that appears in "location info" is constructed using this algorithm.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<b><i>Obligation Information</i></b>	Optional	<p>We use the term "obligation information" to describe the basic information that appears throughout the system to describe an obligation. The data that appears in "obligation information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the system default "obligation information".</p> <p>Note: This algorithm may be further overridden by an "obligation information" plug-in on the obligation type. Refer to <a href="#">Obligation Type – Algorithms</a> for how</p>

		<p>algorithms of this type are used.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Online Bill Display</i>	Optional	<p>This algorithm constructs a PDF that contains the image of a bill. This algorithm is executed when the Display Bill button is clicked on the <a href="#">Bill</a> page. Refer to <a href="#">Technical Implementation of Online Bill Image</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Online Letter Image</i>	Optional	<p>This algorithm constructs a PDF that contains the image of a letter. This algorithm is executed when the Display Letter button is pressed on <a href="#">Customer Contact - Main</a>. Refer to <a href="#">Technical Implementation of Online Letter Image</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Override Proration Factors</i>	Optional	<p>This algorithm is only used if your organization has unusual rate proration requirements that necessitate the overriding of the base package proration logic. For example, you may have certain rate components whose charges should never be prorated. Refer to <a href="#">Overriding Proration Factors</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Override Seasonal Proration</i>	Optional	<p>This algorithm is only used if your organization has unusual method of determining the seasons for your rate components. For example, you may determine the seasonal boundaries for a rate component based on the scheduled meter read date associated with the bill cycle. Refer to the description of the seasonal attributes for a <a href="#">rate component</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Payment Amount Calculation</i>	Required	<p>This algorithm is executed to calculate the amount of an automatic payment for a bill for an account with an active auto pay option. Refer to <a href="#">How And When Are Automatic Payments Created</a> for more information on automatic payments. This algorithm is also executed to default the amount of a manually added payment. Refer to <a href="#">How To Add A New Payment Event</a> for more information on adding a payment manually.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Payment Information</i>	Required	<p>We use the term "payment information" to describe the basic information that appears throughout the system to describe a payment. The data that appears in "payment information" is constructed using this algorithm.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Person Information</i>	Required	<p>In most parts of the system, a person's <b>Main</b> name is displayed to describe a person. However, several transactions do not use this method. Rather, these transactions call the algorithm that's plugged into this spot to construct the person's name. Refer to the description of the <b>Alternate Representation</b> flag on the <b>Main</b> tab for a list of these transactions and for the rationale behind this algorithm.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Person Name Validation</i>	Required	<p>The format of names entered on <a href="#">Person - Main</a> is validated using this algorithm.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>

## Defining Taxpayer Languages

As described under [Defining Languages](#), you define the language in which each user sees the system. In addition to defining each user's language, the system allows you to define each taxpayer's preferred language. For example, one taxpayer can receive bills in English whereas another taxpayer could receive their bills in Chinese.

Each taxpayer's language is defined by the [language code](#) on their [person record](#). Bills, adjustments and other system-generated records will then be done in the language of the main taxpayer of the account. In addition, the language code is passed on to all taxpayer-facing interfaces, such as letter requests and bill print.

**Note.** You can define Rates in multiple languages – when a bill is generated, the line-item descriptions are generated and stored in the account's main taxpayer's language of choice. Any one who subsequently views these bills can only see the descriptions in that language.

**Note.** To support bills and other correspondence, you must also provide translations of standard bill stock and letters. This must be handled by your printing software vendor.

## Defining Accounting Calendars

The accounting calendar determines the accounting period to which a financial transaction will be booked. The following points describe how the system determines a financial transaction's account period:

- Every financial transaction references an accounting date and its obligation
- Every obligation references an obligation type
- Every obligation type references a GL division
- Every GL division references an accounting calendar
- The accounting calendar contains the cross reference between the accounting date specified on the financial transaction and related accounting period in your general ledger

**Warning!** This information must be the same as the information in your financial database.

To add or review an accounting calendar, choose **Admin Menu, Accounting Calendar**.

### Description of Page

Enter a unique **Calendar ID** and **Description** for the calendar.

Enter the **Number Of Periods** for the calendar. Don't count the adjustment period, if you use one, or any special "system" periods.

Specify the **Fiscal Year**, each **Accounting Period** in that year, a **Period Description**, the **Begin Date** and the **End Date**.

When you enter begin and end dates, you can define monthly calendar periods or any fiscal period that matches your accounting calendar (weekly, bimonthly) as long as the begin and end dates of successive periods do not overlap. Every day of the year must be included in a period; do not leave gaps between period dates.

For each fiscal period, enter the **Open From Date** and **Open To Date**. These dates define when that particular business dates are open for posting financial transactions to that fiscal period. For example, you might calculate a bill on Sept 1 for usage recorded on August 31. To post this financial transaction in the August period, you must keep it open through Sept 1.

As time passes, you will need to return to this transaction to manually enter ensuing years. You can enter several years at a time or incorporate the task into end-of-year system maintenance.

### Where Used

Follow this [link](#) to open the data dictionary where you can view the tables that reference [CI\\_CAL\\_GL](#)

## Defining General Ledger Divisions

There are two types of divisions referenced in the system: a division and a GL division. This is a rather powerful structure, but it can be confusing.

- General Ledger divisions typically comprise individual entities (e.g., companies) in your general ledger. You must set up a GL division for each such entity. The GL division's sole purpose in the system is to define the accounting period associated with financial transactions linked to obligations associated with the GL division (obligations are associated with GL divisions via their obligation type). The system cares about accounting periods in order to prevent a user from booking moneys to closed periods. It also uses accounting periods when it produces the flat file that contains the consolidated journal entry that is interfaced to your general ledger (refer to [The GL Interface](#) for more information).

**Note.** When determining how many GL Divisions you need, be sure to consider your general ledger and how your chart-of-accounts is structured. You will typically have one GL division for each "company" in your general ledger.

- A division is typically associated with a jurisdiction. The definition of a jurisdiction is a geographic-oriented entity with unique business rules. You must set up a division for each jurisdiction in which you conduct business.

Refer to [Setting Up Divisions](#) for information about divisions.

To define a general ledger division, select choose **Admin Menu, General Ledger Division**.

### Description of Page

Enter a unique **GL Division** for the general ledger division.

Enter a **Description** of this general ledger division.

Define the accounting **Calendar ID** that controls how to convert an FT's accounting date into an accounting period. Refer to [Defining Accounting Calendars](#) for more information.

You may define a **Currency Code** for the GL division. Note that the system does not use this currency code.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_GL\\_DIVISION](#).

## Defining Banks & Bank Accounts

---

The topics in this section describe how to maintain your implementation's bank accounts.

#### Contents

[Bank - Main](#)

[Bank - Bank Account](#)

### Bank - Main

To add or review Banks choose **Admin Menu, Bank**.

#### Description of Page

Enter a unique **Bank Code** and **Description** for the bank.

The **Bank Accounts** collection displays the bank accounts currently linked to this bank code. Use the drill down button to view more details or to modify the bank account details. Alternatively, you may navigate to the Bank Account tab and scroll to the desired bank account.

### Bank - Bank Account

To add or review Bank Accounts for a Bank, choose **Admin Menu, Bank** and then navigate to the **Bank Account** tab.

#### Description of Page

Use the **Bank Accounts** tab to define the attributes of each bank account. For each account, enter the following information:

- Enter a **Bank Account Key** to identify an Account at a Bank. You may have more than one account at a given bank, and you may have accounts at more than one bank. This code will allow the system to easily identify a specific account.
- Enter a **Description** to appear on prompt lists, inquiries, and reports.
- Enter the **Account Number**, **Check Digit** and if needed, the **Branch ID** of the bank where the account is held.
- Enter the **Currency Code** for the currency in which the account is denominated.
- Use **DFI ID** to define the Depository Financial Institution ID that is interfaced to the automatic payment-processing agent as part of the automatic payment interface.

- Enter the **Distribution Code** to be used for cash GL distributions when a payment is frozen or canceled.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_BANK\\_ACCOUNT](#).

## To Do Lists Addendum

---

This section is an addendum to the general [To Do Lists](#) chapter. This addendum describes the To Do functionality that is specific to Oracle Enterprise Taxation Management.

### Contents

[Assigning A To Do Role](#)  
[System To Do Types](#)

## Assigning A To Do Role

As described in [To Do Entries Reference A Role](#), each To Do entry requires a role. To Do entries created in Oracle Enterprise Taxation Management may attempt to assign a role based on an account management group or division if it is applicable to the type of data related to the To Do entry.

As described in [The Big Picture of To Do Lists](#), users are informed that something requires their attention by entries that appear in a To Do List. For example, consider what happens when billing can't find appraisal information for a property:

- The billing process creates a bill segment that is in error (appraisal information cannot be found).
- This bill segment that's in error, in turn, triggers the creation of a To Do entry.
- The To Do entry is assigned a role. A role is one or more users who can look at / work on the entry.
- When users view a To Do List, they only see entries addressed to roles to which they belong.

You can optionally use account management groups (AMG) to define the respective role to be assigned to To Do entries that are associated with an account and To Do type. For example, you can create an AMG called **Credit Risks** and assign this to accounts with suspect credit. Then, whenever an account-oriented To Do entry is created for such an account, it will be assigned a role based on the **Credit Risks** AMG. Refer to [Setting Up Account Management Groups](#) for more information.

By assigning an AMG to an account, you are telling the system to address this account's To Do list entries to the roles defined on the AMG (note, each To Do type can have a different role defined for it on an AMG).

You can optionally use division to define the respective role to be assigned to To Do entries that are associated with an account and To Do type. Refer to [Setting Up Divisions](#) for more information.

A **To Do Pre-Creation** installation options plug-in is provided to determine the appropriate To Do Role for an account based on AMG and division setup. If plugged in, the logic to determine To Do role for an account is performed whenever a To Do entry is created. Refer to [C1-TDCR-DFRL](#) for further details on how this plug-in works.

Refer to [To Do Entries Reference A Role](#) for the details of how an initial role is assigned to To Do entries.

## System To Do Types

**List of available To Do types.** The To Do types available with the product may be viewed in the [application viewer](#)'s [To Do type](#) viewer. In addition if your implementation adds To Do types, you may [regenerate](#) the application viewer to see your additions reflected there.



# Defining Financial Transaction Options

Bills, payments and adjustments share one very important trait – they affect how much your taxpayers owe. This section explains the financial design of the system and describes how to set up the tables that control the financial impact of these transactions.

## Contents

- [The Financial Big Picture](#)
- [Obligation Type Controls Everything](#)
- [Tender Management](#)
- [Automatic Payment Options](#)
- [Payment Advices](#)
- [Payment Event Distribution](#)
- [Cancel Reasons](#)
- [Miscellaneous Financial Controls](#)
- [Payables Cash Accounting](#)
- [Open Item Accounting](#)
- [Fund Accounting](#)
- [Defining Bill Cycles and Bill Periods](#)
- [Other Financial Transaction Topics](#)

## The Financial Big Picture

---

This section provides an overview of the relationship between an account and the various financial transactions that influence how much a taxpayer owes.

**Warning!** If your organization practices cash accounting for payables (i.e., you only pay the taxing authority when you get paid), refer to [Payables Cash Accounting](#). If your organization practices open-item accounting (i.e., payments must be matched to bills), refer to [Open Item Accounting](#).

## Contents

- [Assessments Overview](#)
- [Penalty and Interest](#)
- [Bills, Payments & Adjustments](#)
- [Bill Details](#)
- [Payment Details](#)
- [Adjustment Details](#)
- [Current Amount versus Payoff Amount](#)
- [Preventing Obligation Balances And The GL From Being Impacted Until Bill Completion](#)
- [Forcing The Freeze Date To Be Used As The Accounting Date](#)

## Assessments Overview

An assessment is a tax liability financial transaction associated with an obligation and represents the legal presentation of a tax liability to a taxpayer. There are different types of assessments, based on how and why the liability was created, and the assessment type can apply differentiated processing rules, such as different penalty and interest calculation rules or different collections rules.

Most filing periods only have one assessment created that captures any applicable penalty, interest, or fees. However, it is possible for additional assessments to be created within a filing period.

Return and bill based tax types differ in how assessments get created and processed. The following sections discuss these concepts.

## Contents

- [Return Based Taxes](#)
- [Bill Based Taxes](#)
- [Group Financial Transactions](#)
- [Effective Date](#)

## Return Based Taxes

Return based taxes are either expected or event based. Return based taxes represent taxpayers that have an expected obligation to file a *Return* for a filing period. Return based taxes can also be based on events such as taxpayers filing an excise tax. Income tax, withholding tax, and sales tax are examples of return based taxes.

For expected return-based tax types, the obligation is associated with a filing period as it tracks whether a filing obligation has been met for that period, and groups the assessments and financial transactions related to it. For most tax types, a filing period may be associated with a single type of obligation. For example, an active monthly Sales Tax filer has a single obligation for each month of the year. However, some tax types may require multiple filing obligations (of different types) over the same period of time. For example, a Withholding Tax type may have monthly Withholding Tax Returns as one obligation type, plus an annual Reconciliation Return and an Annual Taxpayer Withholding Detail statement as other obligation types.

Most return-based tax liability is created through the original self-assessment reported by a taxpayer on their tax return. There are two common models for self-assessment:

- **Full Self-Assessment.** This is common in the US. In full self-assessment, a taxpayer reports their income details on their tax return, calculates their tax per the tax return instructions, and files (and pays or expects a refund) by the due date of the filing period. If the tax return has errors or is changed during processing, the taxpayer receives an adjustment notice, but never receives an assessment notice.
- **Partial Self-Assessment.** This is common in taxes modeled after the British tax system. Like full self-assessment, the taxpayer reports their income tax details on a tax return. However, the tax return does not include instructions for calculating tax. When the tax return is processed, a Notice of Assessment is generated and sent to the taxpayer. The Notice of Assessment is a legal document informing the taxpayer of their tax obligation, and the due date for paying it.

The assessment established by the taxpayer's initial tax return is called the original assessment. If the taxpayer is later audited, and their income is increased (such that they now owe more tax), the additional tax established by the audit is part of an audit assessment. The incremental tax from the audit will be subject to different business rules, such as different rates and rules for calculating penalty and interest, and different collections notices or processes. It is also possible for a change to a tax return to result in a decrease in tax (and therefore a refund).

## Bill Based Taxes

Bill based taxes are billed on a predefined schedule. Property tax, periodic licensing fees, and metered oil/resource taxes are examples of bill-based taxes.

For bill-based taxes, the system has all of the necessary information to initiate assessments. The system creates assessments on a schedule and a single assessment exists for each billing period. If a taxpayer disputes a bill, the bill is cancelled and rebilled without creating an incremental assessment.

## Group Financial Transactions

As financial transactions are created for an obligation, often they are related to a specific liability such as an assessment, which is also a financial transaction (associated with an adjustment or a bill segment). For example, a payment may be made for a specific assessment. Penalties, interest and fees are often levied toward a specific assessment. Associating these subsequent financial transactions with the appropriate assessment FT ensures accurate penalty and interest calculations and accurate views of the balance of each assessment associated with an obligation.

The system provides a means for your implementation to group financial transactions (via the Group FT ID). When creating a financial transaction that should be associated with a given assessment, set the group FT id to the FT id of the assessment. The following example shows the usage of the Group FT Id field.

FT ID	FT Type	Parent ID	Amount	Group FT ID
123456	Adjustment	INC-TAX	2000	123456
483940	Adjustment	PENALTY	50	123456
864728	Adjustment	INTEREST	10	123456
637483	Payment	6387362	<2060>	
345678	Adjustment	INC-TAX	1000	345678
247389	Adjustment	INC-TAX	<50>	345678
748627	Payment	748765	<950>	

Tax FT references itself

Not every FT is linked to an assessment

Correction to a return may be grouped with an existing assessment

The original assessment FT (sometimes referred to as the "header FT") references its own FT id in the Group FT field. This allows system functionality to easily identify the assessment financial transaction.

## Effective Date

The effective date is the effective date of the payment, bill, or adjustment. For more information about effective dates and payment dates, see [Payment Date and Effective Date for Payment Events](#). The effective date is used in calculating penalties and interest. For more information, refer to [The Big Picture of Penalty and Interest](#).

## Penalty and Interest

Calculating penalty and interest (P&I) for delinquent assessments is an important element of a tax authority's business.

Refer to [The Big Picture of Penalty and Interest](#) for details about this functionality.

## Bills, Payments & Adjustments

The following points are high level descriptions of the various financial transaction supported

- For bill based taxes, many bills are produced for an account over time. For more information about a bill, see [Bill Details](#).
- Over time, many payments may be applied to an account's debt. For more information about a payment, see [Payment Details](#).
- Over time, the debt that is stored on an account's obligation(s) may be adjusted. For more information about an adjustment, see [Adjustment Details](#).

Note that the system maintains debt on each individual obligation for an account. An account's debt is the sum of its obligations' debt.

### Bill Details

The following points highlight important concepts related to bills:

- A bill is produced for an account. Over time, many bills may be produced for an account.
- Bills contain bill segments. A bill segment is created to levy charges for a bill-based obligation. You may configure your system to generate a single bill for a single obligation. For example you may generate one bill for your property tax obligation's charges. You may also include bill segments for multiple obligations in a single bill. For example you may generate a bill that includes bill segments for all the personal property obligations.
- Bill segments contain calculation details. A bill segment contains information showing how the segment was calculated and how it should be printed on the taxpayer's bill.
- A bill segment has a financial transaction. A bill segment has a related financial transaction. A financial transaction contains the financial effects of the bill segment on the obligation's current and payoff balances and on the general ledger.
- Canceling a bill cancels the financial transaction. If the bill segment is eventually cancelled, another financial transaction will be linked to the bill segment to reverse its original financial transaction. The cancellation financial transaction appears on the next bill produced for the account as a bill correction.

### Payment Details

The following points highlight important concepts related to payments:

- A payment amount is ultimately directed towards an obligation via a payment segment.
- If an account has multiple obligations that are in debt, the various payment segments created for the obligations may be grouped into a single Payment for the account.

- A payment segment has a related financial transaction. A financial transaction contains the financial effects of the segment on the obligation's current and payoff balances and on the general ledger.
- Canceling a payment cancels the financial transaction. If the payment is eventually cancelled, another financial transaction will be linked to the related payment segment(s) to reverse their financial effect. The cancellation financial transaction appears on the next bill produced for the account as a negative payment.

A payment cannot be applied to an account's debt without an associated payment event. Refer to [The Big Picture of Payments](#) for more information.

## Adjustment Details

The following points highlight important concepts related to adjustments

- Over time, an obligation may have many adjustments.
- An adjustment has a related financial transaction. The financial transaction contains the financial effects of the adjustment on the obligation's debt and on the general ledger.
- Canceling an adjustment cancels the financial transaction. If the adjustment is eventually canceled, another financial transaction will be linked to the adjustment to reverse its financial effect. The cancellation financial transaction appears on the next bill produced for the account as an adjustment.

## Current Amount versus Payoff Amount

A financial transaction contains two amount attributes: payoff amount and current amount. These attributes allow you to keep track separately of amounts related to how much the taxpayer owes.

- Current amount contains how much the taxpayer THINKS THEY OWE. In other words, this is the amount that affects the taxpayer's balance. It is what the taxpayer owes with respect to collections and is the amount that penalty and interest is calculated on.
- Payoff amount contains how much the taxpayer REALLY OWES. This is the amount posted to the general ledger.

For most financial transactions, these values are the same. There are a small number of cases where this amount may be different. One example is a charitable contribution. If a taxpayer makes a charitable contribution when paying their tax liability, the payment or adjustment associated with the contribution credit should affect the payoff amount and the general ledger (because this amount is actually cash in hand). However, it should not affect the current amount because the contribution is not paying off any debt incurred and should not cause the taxpayer's balance to go into credit.

The topics in this section provide more information about these two fields.

### Contents

[What Controls What Gets Booked To Current And Payoff Amount?](#)  
[GL Accounting Information](#)

## What Controls What Gets Booked To Current And Payoff Amount?

As described in [Bill Details](#), every bill segment has a sibling financial transaction. The financial transaction defines the bill segment's affect on current and payoff amounts due. The system populates these two fields as per the Financial Transaction Algorithm defined on the bill segment's bill segment type.

For more information, refer to [Billing – Current Balance versus Payoff Balance](#) and [Defining Bill Segment Types](#).

As described in [Payment Details](#), every payment segment has a sibling financial transaction. The financial transaction defines the payment segment's affect on current and payoff amounts due. The system populates these two fields as per the Financial Transaction Algorithm defined on the payment segment's payment segment type.

For more information, refer to [Payment – Current Balance versus Payoff Balance](#) and [Setting Up Payment Segment Types](#).

As described in [Adjustment Details](#), every adjustment has a sibling financial transaction. The financial transaction defines the adjustment's affect on current and payoff amounts due. The system populates these two fields as per the Financial Transaction Algorithm defined on the adjustment's adjustment type.

For more information, refer to [Adjustments – Current Balance versus Payoff Balance](#) and [Setting Up Adjustment Types](#).

## GL Accounting Information

Be aware that if payoff amount is non-zero, the financial transaction has general ledger detail lines.

The effect on your GL is controlled by the financial transaction algorithm defined on your bill segment and payment segment types.

Refer to [The GL Interface](#) for how GL account information is interfaced to the general ledger.

## Preventing Obligation Balances And The GL From Being Impacted Until Bill Completion

It's important to understand that when any type of financial transaction is **frozen**, the related obligation's balance is affected. For example:

- When a payment is **frozen**, the taxpayer's balance is reduced.
- When an adjustment is **frozen**, the taxpayer's balance is impacted.
- When a bill segment is **frozen**, the taxpayer's balance is increased (typically).

For payments, there is no issue. However, for bill based tax types, you may NOT want the taxpayer's balance to be impacted until the bill is completed. Consider the following:

- If a taxpayer has multiple obligations that should be presented on the same bill, it's possible for one of the obligations to have a bill segment that's in **error** and the other obligation's bill segment to be **frozen**.
- The **frozen** bill segment will impact the taxpayer's balance and the general ledger. This is because a financial transaction is marked for **interface** to the general ledger when it is frozen. This can be problematic if you have a long period between FT freeze and bill completion (you could impact the general ledger but not impact the taxpayer's balance).

If this is unacceptable, you can setup the system to not allow certain types of FT's to be frozen until the bill is completed. This means that neither the taxpayer's balance nor the general ledger will be impacted until bill completion time. To do this:

- Choose the **Freeze At Bill Completion** option on [Installation Options - Billing](#).
- Determine if any of your [adjustment types](#) are ones that would be included on a bill-based tax bill. Select **Freeze At Bill Completion** for those that should not impact the taxpayer's balance or the general ledger until the next bill is completed. Select **Freeze At Will** for those that should impact the taxpayer's balance and the GL when they are frozen.

Please be aware of the following in respect of the **Freeze At Bill Completion** options:

- If you turn on **Freeze At Bill Completion** on [Installation Options - Billing](#):
  - Users will not be allowed to freeze bill segments online.
  - Any background process created for batch billing should not freeze bill segments until all segments on a bill are error free.
  - Bill segments will exist in the **freezable** state until the bill is **completed**.
- If you turn on **Freeze At Bill Completion** on [Adjustment Type - Main](#):
  - Users will not be allowed to freeze adjustments of this type online.
  - Background processes that create adjustments will not freeze this type of adjustment. Rather, the adjustments will be frozen when the next bill is completed.
  - Adjustments of this type will therefore exist in the **freezable** state until the next bill is **completed**.

**Alerts highlight freezable FT's.** Please be aware that messages appear in the [Account Information - Financial Information Zone](#) and in the [Dashboard - Financial Information Zone](#) to highlight the existence of freezable financial transactions.

Please be aware of the following in respect of the **Freeze At Will** options:

- If you turn on **Freeze At Will** on [Installation Options - Billing](#):
  - Users will be allowed to freeze bill segments online.
  - Any background process created for batch billing should freeze bill segments when the individual segment is error-free.
  - Bill segments will exist in the **frozen** state regardless of whether the bill is completed.
  - The **frozen** bill segment's FT will be interfaced to the GL when the interface next runs.
  - All adjustment types must also be set to **Freeze At Will** (otherwise they wouldn't get frozen).



- If you turn on **Freeze At Will** on [Adjustment Type - Main](#):
  - Users will be allowed to freeze adjustments of this type online.
  - Background processes that create adjustments will freeze this type of adjustment.
  - Adjustments of this type will exist in the **frozen** state prior to bill completion.
  - The **frozen** adjustment's FT will be interfaced to the GL when the interface next runs.

## Forcing The Freeze Date To Be Used As The Accounting Date

Every financial transaction references an accounting date. The accounting date controls the accounting period to which the financial transaction is booked as described below:

- Every financial transaction references an accounting date and an obligation
- Every obligation references an obligation type
- Every obligation type references a GL division
- Every GL division references an [accounting calendar](#)
- The accounting calendar contains the cross reference between the accounting date specified on the financial transaction and the related accounting period in your general ledger

The accounting date is populated on financial transactions when they are initially generated. The following points describe the source of the accounting date:

- The user who creates or cancels a bill segment online defines the accounting date as part of the generation / cancel dialog (note, the current date defaults).
- The user who creates or cancels an adjustment online defines the accounting date as part of the generation / cancel dialog (note, the current date defaults).
- Payments are unusual in that their financial transaction is only created when they are frozen (rather than when the payment is first distributed amongst the account's obligations). At payment freeze time, the accounting date is set to the current date.

For payments, there is no issue because the accounting date is only populated on the financial transaction when a payment is frozen. However, for bill segments and adjustments, your business practice may dictate that the freeze date should be used as the accounting date rather than the original accounting date. Alternatively, your business practice may dictate that the accounting date that's originally stamped on bill segments / adjustments should be used (unless this associated period is closed at freeze time). It's really a question of the interpretation of the local accounting rules. After you've decided on your approach, populate the **Accounting Date Freeze Option** on [Installation Options - Billing](#) with one of the following values:

- Choose **Always change** if the accounting date on your financial transactions should be populated with the freeze date (i.e., the current date when the financial transaction is frozen).
- Choose **Change if period is closed** if the accounting date defined when the financial transaction is generated should be used (unless the associated accounting period is closed).

Please be aware of the following in respect of your choice:

- If you choose **Always change**:



- When a user freezes a bill segment online, they will be prompted to supply an accounting date. The current date will default, but the user can override this value.
- When a user freezes an adjustment online, they will be prompted to supply an accounting date. The current date will default, but the user can override this value.
- Any batch billing process should use the current business date as the accounting date on bill segments that it freezes.
- Also note, if you have chosen the **Freeze At Bill Completion Bill Segment Freeze Option** on the [installation record](#), bill segments and certain types of adjustments are frozen when a bill is completed. This means that the accounting date on the related financial transactions will be set to the completion date (because the completion date is the freeze date with this setting). Refer to [Preventing Obligation Balances And The GL From Being Impacted Until Completion](#) for more information.
- If you choose **Change if period is closed**:
  - When a user freezes a bill segment online, they will only be prompted to supply an accounting date if the related accounting period is closed (because the accounting period closes after the bill segment is generated but before it's frozen). The current date will default, but the user can override this date.
  - When a user freezes an adjustment online, they will only be prompted to supply an accounting date if the related accounting period is closed (because the accounting period closes after the adjustment is generated but before it's frozen). The current date will default, but the user can override this date.

## Obligation Type Controls Everything

The previous section illustrated three important concepts:

The true financial impact of the three financial events - bills, payments, adjustments - is at the obligation level, not at the account level. This means that bills and payments are meaningless on their own. It's the obligations' bill segments, payment segments and adjustments that affect how much a taxpayer owes.

- Every bill segment, payment segment, and adjustment has a related financial transaction. These financial transactions contain the double-sided journal entries that will be interfaced to your general ledger. They also contain the information defining how the taxpayer's debt is affected by the financial event (i.e., current amount and payoff amount).
- A single bill can contain many bill segments, each of which may have a different frequency.

You control the financial effects of the various financial events using a single field on the obligation. This field is called the **Obligation Type**. In this section, we describe many of the tables that must be set up before you can create an obligation type.

**Foreshadowing.** You will notice that we don't explain how to set up obligation types in this section. This is because obligation type controls numerous aspects of an obligation's behavior in addition to its financial behavior. The non-financial aspects are discussed in later chapters. It's only after you have set up all of the control tables in this manual that you'll be able to finally define your obligation types. Refer to [Setting Up Obligation Types](#) for more information.

**Warning!** Take the time to define how you will record the various financial events in your general ledger before you attempt to set up these control tables. If you have simple accounting needs, this setup process will be straightforward. However, if you sell many services and use sophisticated accounting, this setup process will require careful analysis.

## Contents

- [Setting Up Divisions](#)
- [Setting Up Revenue Classes](#)
- [Setting Up Debt Categories](#)
- [Setting Up Debt Category Priorities](#)
- [Setting Up Distribution Codes](#)
- [Setting Up Billable Charge Templates](#)
- [Defining Bill Segment Types](#)
- [Setting Up Payment Segment Types](#)
- [Designing And Defining Adjustment Types](#)

## Setting Up Divisions

There are two types of divisions referenced on an obligation type: a division and a GL division.

- General Ledger divisions typically comprise individual entities (e.g., companies) in your general ledger. You must set up a GL division for each such entity. The GL division's sole purpose in the system is to define the accounting period associated with financial transactions linked to obligations associated with the GL division (obligations are associated with GL divisions via their obligation type). The system cares about accounting periods in order to prevent a user from booking moneys to closed periods. It also uses accounting periods when it produces the flat file that contains the consolidated journal entry that is interfaced to your general ledger (refer to [The GL Interface](#) for more information).
- A division is used to separate business rules and can often be associated with a jurisdiction. The definition of a jurisdiction is a geographic-oriented entity with unique business rules. Another example of where you may use separate divisions is when your tax authority is responsible for other types of debt in the system, such as state university debts or child support debts. You must set up a division for each segmentation in your authority where business rules may differ.

Division is also referenced on obligation, location and account.

- The division on obligation is actually part of the obligation's obligation type. Because obligation type controls many business rules, all business rules that are on the obligation type can be thought of as being defined for a given jurisdiction and obligation type combination. Refer to [Defining Obligation Types](#) for more information.
- The division on location defines the jurisdiction in which the location is located. This jurisdiction controls the types of obligations that can be associated with the location.
- The division on account when combined with the account's account type defines the jurisdiction that governs financial business rules (e.g., the bill's due date). Refer to [Setting Up Account Types](#) for more information about these rules. The division on account can also play a part in the addressee of To Do entries associated with the account. To assign To Do entries to a role based on the division, simply link the To Do type to the division. Refer to [To Do Entries Reference A Role](#) for more information.

**Note.** Both division and GL division are stored on the financial transactions associated with an obligation. However, only GL division plays a part in [The GL Interface](#). Refer to [Setting Up GL Divisions](#) for information about GL Divisions.

The following topics describe the pages used to maintain a division.

## Contents

[Division - Main](#)

[Division - Characteristics](#)

## Division - Main

To define a division, select choose **Admin Menu, Division**.

### Description of Page

Enter an easily recognizable **Division** and **Description** for the division.

Enter the **Work Calendar** that defines the days on which this division operates. This calendar is used to ensure system-calculated dates (e.g., bill due date, credit and collection event dates, etc.) fall on a workday.

Use the **To Do Roles** scroll area if an account's division influences the role assigned to To Do entries associated with the account. In the collection, define the **To Do Role** to be assigned to entries of a given **To Do Type** that are associated with accounts that reference the **Division**. Refer to [Assigning A To Do Role](#) for more information.

**Note.** Only To Do entries that are account-oriented take advantage of the roles defined for a division.

### Where Used

Follow this link to view the tables that reference [CI CIS DIVISION](#) in the data dictionary schema viewer.

## Division - Characteristics

You can define characteristics for a division. You may need these for reporting purposes or in your algorithms. Refer to [Characteristic Types](#) for more information.

Open **Admin Menu, Division** and navigate to the **Characteristics** tab to maintain a division's characteristics.

### Description of Page

Select a **Characteristic Type** and **Characteristic Value** to be associated with this division. Indicate the Effective Date of the characteristic type and value.

**Note.** You can only choose characteristic types defined as permissible on a division record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

## Setting Up Revenue Classes

Every obligation references an obligation type. Amongst other things, the obligation type defines an obligation's revenue class. The revenue class is used when the obligation's rate books revenue to different GL distribution codes based on the obligation's revenue class.

See [Rate Component – GL Distribution](#) for more information about how revenue class is used to determine the GL revenue accounts referenced on a bill.

To set up revenue classes, choose **Admin Menu, Revenue Class**.

### Description of Page

Enter an easily recognizable **Revenue Class ID** and **Description** for every revenue class.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_REV\\_CL](#).

## Setting Up Debt Categories

Debt Categories are used to categorize financial transactions that are debits. Debt categories may also be assigned to credit financial transactions if the credit is associated with a given type of debit. Refer to [Debt Categories and their Priorities](#) for more information.

To set up a debt category, open **Admin Menu, Debt Category**.

The topics in this section describe the base-package zones that appear on the Debt Category portal.

### Contents

- [Debt Category List](#)
- [Debt Category Actions](#)
- [Debt Category](#)

### Debt Category List

The Debt Category [List zone](#) lists every debt category. The following functions are available:

- Click the [broadcast](#) icon to open other zones that contain more information about the adjacent debt category.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each debt category.

Click the **Add** link in the zone's title bar to add a new debt category.

### Debt Category Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions are available.

## Debt Category

The Debt Category zone contains display-only information about a debt category. This zone appears when a debt category has been broadcast from the Debt Category List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_DEBT\\_CAT](#).

## Setting Up Debt Category Priorities

Debt category priorities allow you to define a priority order of allocating credit financial transactions to debit financial transactions during credit allocation. Refer to [Debt Categories and their Priorities](#) for more information.

To set up a debt category, open **Admin Menu, Debt Category Priority**.

The topics in this section describe the base-package zones that appear on the Debt Category Priority portal.

### Contents

- [Debt Category Priority List](#)
- [Debt Category Priority Actions](#)
- [Debt Category Priority](#)

## Debt Category Priority List

The Debt Category Priority [List zone](#) lists every debt category priority. The following functions are available:

- Click the [broadcast](#) icon to open other zones that contain more information about the adjacent debt category priority.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each debt category priority.

Click the **Add** link in the zone's title bar to add a new debt category priority.

## Debt Category Priority Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions are available.

## Debt Category Priority

The Debt Category Priority zone contains display-only information about a debt category priority. This zone appears when a debt category priority has been broadcast from the Debt Category Priority List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_DEBT\\_CAT](#).

## Setting Up Distribution Codes

Distribution codes simplify the process of generating accounting entries by defining valid combinations of chart of account field values.

Refer to [The Source Of GL Accounts On Financial Transactions](#) for more information about the accounting entries associated with bills, payments and adjustments.

To set up distribution codes, open **Admin Menu, Distribution Code**.

### Description of Page

Enter a unique **Distribution Code** and **Description** for the distribution code.

If this distribution code is a holding account used for payables cash accounting, check the **Use For Cash Accounting** switch, and enter the actual payable **Cash Accounting Code**. The system will transfer the holding amount to this distribution code when the cash event occurs. For more information, refer to [Payables Cash Accounting](#).

Define the **GL Account Algorithm** used by the system when it interfaces financial transactions that reference this distribution code to your general ledger (refer to [GLDL - Create General Ledger Download](#) for more information about the download process). The logic embedded in this algorithm constructs the actual GL account number. If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that constructs your general ledger account number. Click [here](#) to see the algorithm types available for this plug-in spot.

The **Write Off Controls** provides the ability to define configuration to write off debt associated with the distribution code by transferring the written off debt to a separate obligation. You would only do this if you wanted the written off debt to remain visible in the account's balance and you don't want it to remain in the original obligation's balance.

- Define the **Division** and **Obligation Type** of the obligation to which bad debt associated with this distribution code should be transferred at write-off time. Note: only obligation types with a special role of **Write Off** may be selected.
- When the system transfers debt to the write-off obligation defined above, the distribution code defined on this **Division / Obligation Type** will be debited unless you turn on the **Override Switch**. When this switch is turned on, the system overrides the distribution code of the transfer to side of the adjustment with the distribution code associated with the debt being written off. You'd typically turn this switch on for liability distribution codes because you want to debit the original liability account when the debt is written off. Note: if this switch is on the system also overrides the characteristic type / value with the respective value associated with the debt that is being written off.

**Note.** The write off algorithms provided in the base product (as part of overdue processing) do not transfer the debt to a separate **write off** obligation.

Use the **GL Account Details** scroll to define how the system constructs the GL account associated with the distribution code when it interfaces the financial transaction to your general ledger. For each distribution code, enter the following information:

- Enter the **Effective Date** of the following information.
- Define whether, on the **Effective Date**, the following information is **Active** or **Inactive**. The system will only use effective-dated information that is **Active**.
- Enter the **GL Account** that the general ledger uses to process financial transactions tagged with this distribution code.
- By default, the installation is configured to practice **fund accounting**. With this option activated, you can define the **Fund** associated with this distribution code. If your installation options indicate that fund accounting is **not practiced**, the field is not visible.
- Use the grid to define characteristic values for the **Distribution Code**. To modify a characteristic, simply move to a field and change its value. The following fields display:
  - **Characteristic Type**. Indicate the type of characteristic.
  - **Characteristic Value**. Indicate the value of the characteristic.

**Note.** You can only choose characteristic types defined as permissible on the distribution code record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_GL\\_DIVISION](#).

## Setting Up Billable Charge Templates

A user creates a billable charge whenever a taxpayer should be levied an ad hoc charge. For example, if a taxpayer requires a review of their property assessment, you may charge them an administration fee.

**Interfacing billable charges from an external system.** In addition to being entered manually, billable charges can also be interfaced from an external system. You would interface billable charges if your organization provides "pass through" billing services. Refer to [Uploading Billable Charges](#) for more information.

A billable charge must reference an obligation. This obligation behaves just like any other obligation:

- **Bill segments are created for the obligation.** Whenever billing is performed for an account with billable charge obligations, the system creates a bill segment for each unbilled billable charge.
- **Payments are distributed to the obligation.** Payments made by an account are distributed to its billable charge obligations just like any other obligation.
- **Overdue debt is monitored.** The credit and collections process monitors billable charge obligations for overdue debt and responds accordingly when overdue debt is detected.

**Rates can be applied to billable charges.** Billable charges can be connected to an obligation that also specifies a rate. The rate will be applied and lines added to the bill segment after the billable charge lines are added. For example, a rate can insert flat charges or be applied to service quantities associated with the billable charge.

Refer to [How To Create An Ad-hoc Bill](#) for instructions describing how to create a bill for a billable charge outside of the normal bill creation process.

Billable charge templates exist to minimize the effort required to create a billable charge for a taxpayer. A billable charge template contains the default bill lines, amounts and distribution codes used to levy a one-off charge.

The information on the template may be overridden by a user when the billable charge is created.

**Templates aren't required.** A billable charge can be created without a template for a truly unexpected charge.

After setting up the billable charge templates, you must indicate the obligation types that can use each template. Obviously, only **billable charge** obligation types (as defined on the obligation type's special role) will reference billable charge templates.

## Contents

- [Billable Charge Template - Main](#)
- [Billable Charge Template - Line Characteristics](#)
- [Billable Charge Template - RQ Details](#)

## Billable Charge Template - Main

Open **Admin, Billable Charge Template** to define your billable charge templates.

### Description of Page

Enter a unique **Billable Charge Template ID** and **Description** for the billable charge template.

Use **Description on Bill** to define the verbiage that should print on the taxpayer's bill above the billable charge's line item details.

Use **Currency Code** to define the currency in which the billable charge's amounts are expressed.

Use the grid to define the line item details associated with the billable charge (note, the **Total Line Amount** field is automatically calculated. It is the sum of the **Charge Amount** on each of the Line Sequence items). The following fields are required for each entry in the grid.

<b>Sequence</b>	Line sequence controls the order in which the line items appear on the bill segment.
<b>Description on Bill</b>	Specify the verbiage to print on the bill for the line item.
<b>Charge Amount</b>	Specify the default amount to charge for the line item.
<b>Show on Bill</b>	Turn this switch on if the line item should appear on the taxpayer's printed bill. It would be very unusual for this switch to be off.



**Appears in Summary**

Turn this switch on when the amount associated with this line also appears in a summary line.

**Memo Only, No GL**

Turn this switch on when the amount associated with this line does not affect the GL (or the total amount owed by the taxpayer).

**Distribution Code**

Specify the default distribution code associated with this line item.

If you use the drill down button on the left most column in the grid, you will be taken to the Line Characteristics tab with the selected line displayed.

For more information about creating a billable charge, refer to [Maintaining Billable Charges](#). For more information about billing billable charges, refer to [How To Create An Ad-hoc Bill](#).

**Billable Charge Template - Line Characteristics**

Open **Admin, Billable Charge Template, Line Characteristics** to define your billable charge templates line characteristics.

**Description of Page**

The **Line Sequence** scroll defines the billable charge template line to which you wish to assign characteristic values.

The following fields display:

**Characteristic Type**

The type of characteristic.

**Characteristic Value**

The value of the characteristic.

**Billable Charge Template - RQ Details**

Open **Admin, Billable Charge Template, RQ Details** to define your billable charge templates service quantities.

**Description of Page**

The following fields display:

**Sequence**

Specify the sequence number of the RQ.

**UOM**

Select the unit of measure of this RQ. One or more of UOM, TOU, or RQ identifier must be selected.

**TOU**

Select the time of use period.

**RQ Identifier**

Select the RQ identifier.

**Rate Quantity**

Specify the number of units of this rate quantity.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_B\\_CHG\\_TMPLT](#).

## Defining Bill Segment Types

Every obligation references an obligation type. Amongst other things, the obligation type references a bill segment type. The bill segment type controls how bill segments and their related financial transactions are created.

**Warning!** We strongly recommend understanding the concepts described in [The Big Picture of Billing](#) before setting up your bill segment types.

### Setting Up Bill Segment Types

To set up bill segment types, open **Admin Menu, Bill Segment Type**.

#### Description of Page

Enter an easily recognizable **Bill Segment Type** and **Description** for every type of bill segment.

For each bill segment type, define the **Create Algorithm**. The logic embedded in this algorithm creates the bill segment.

If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that creates a bill segment in the appropriate manner. Click [here](#) to see the algorithm types available for this plug-in spot.

**Warning!** The **BS Create Algorithm** is a very important field as it controls how the system creates bill segments. There are some restrictions in respect of the values of certain fields on the obligation type and the bill segment algorithm used on an obligation type.

For each bill segment type, define the **Financial Algorithm**. The logic embedded in this algorithm constructs the financial transaction associated with the bill segment.

If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that constructs the bill segment financial transaction in the appropriate manner. Click [here](#) to see the algorithm types available for this plug-in spot.

For more information about current and payoff amounts, refer to [Current Amount versus Payoff Amount](#).

Define the **Pre-creation Algorithm**. The logic embedded in this algorithm retrieves detailed information needed to bill the bill segment.

If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that retrieves details needed for billing in the appropriate manner. Click [here](#) to see the algorithm types available for this plug-in spot.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI BILL SEG TYP](#).

## Setting Up Payment Segment Types

Every obligation references an obligation type. Among other things, the obligation type references a payment segment type. The payment segment type controls how payment segments and their related financial transactions are created. To set up payment segment types, open **Admin Menu, Payment Segment Type**.

### Description of Page

Enter an easily recognizable **Payment Segment Type** and **Description** for every type of payment segment.

For more information about the source of the distribution codes on financial transactions, see [The Source Of GL Accounts On Financial Transactions](#).

For each payment segment type, define the **Payment Segment Fin Algorithm**. The logic embedded in this algorithm constructs the actual financial transaction associated with the payment segment. Refer to [Examples of Common Payment Segment Types](#) for examples of how algorithms are used on common payment segment types.

If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that constructs the payment segment financial transaction in the appropriate manner. Click [here](#) to see the algorithm types available for this plug-in spot.

For more information about current and payoff amount, see [Current Amount versus Payoff Amount](#).

## Examples of Common Payment Segment Types

The following table shows several classic payment segment types used by many organizations:

Payment Segment Type	Payment Segment Financial Transaction Algorithm
Normal payment (if you practice accrual accounting). Refer to <a href="#">Accrual versus Cash Accounting</a> for more information.	<a href="#">Payoff = Pay Amount / Current = Pay Amount (no cash accounting)</a>
Normal payment (if you practice cash accounting). Refer to <a href="#">Accrual versus Cash Accounting</a> for more information.	<a href="#">Payoff = Pay Amount / Current = Pay Amount (plus Cash Accounting)</a>
Charity payment	<a href="#">Payoff = 0 / Current = Pay Amount (the GL is affected)</a>

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI PAY SEG TYPE](#).

## Designing And Defining Adjustment Types

An obligation's debt may be changed with an adjustment. Every adjustment must reference an adjustment type. The adjustment type contains a great deal of information that is defaulted onto the adjustment, including whether the adjustment amount is calculated. It also controls many business processes associated with the adjustment. The topics in this section describe how to design and set up adjustment types.

### Contents

- [What Do Adjustment Types Do?](#)
- [Setting Up Adjustment Types](#)
- [Setting Up Adjustment Type Profiles](#)
- [The Big Picture Of Adjustment Approval](#)
- [Setting Up Approval Profiles](#)

### What Do Adjustment Types Do?

An adjustment type contains the business rules that govern how its adjustments are managed by the system. Please refer to [The Big Picture Of Adjustments](#) for a complete description of how adjustment types impact the lifecycle of adjustments.

### Setting Up Adjustment Types

The topics in this section describe how to set up adjustment types.

**When a new adjustment type is added.** When you introduce a new adjustment type, you must update one or more adjustment profiles with the new adjustment type. Why? Because adjustment profiles define the adjustment types that may be levied on service agreements (adjustment profiles are defined on SA types). If you don't put the adjustment type on an adjustment profile, the adjustment type can't be used on any adjustment.

### Contents

- [Adjustment Type - Main](#)
- [Adjustment Type - Adjustment Characteristics](#)
- [Adjustment Type - Algorithms](#)
- [Setting Up Calculated Adjustment Types](#)

#### Adjustment Type - Main

To set up adjustment types, open **Admin Menu, Adjustment Type**.

#### Description of Page

Enter a unique **Adjustment Type ID** and **Description** for the adjustment type.

The **Adjustment Amount Type** indicates whether the adjustment amount is calculated or not. Select **Calculated Amount** when you want to use a rate to perform calculations to generate the adjustment amount otherwise select **Non-Calculated Amount**. Refer to [Setting Up Calculated Adjustment Types](#) for more information about calculated adjustments.

Enter the **Distribution Code** that references the GL account associated with the adjustment. For example, if this adjustment type is used to levy a charge for a bad check, the distribution code would reference the revenue account to which you associate such revenue. Note, the offsetting distribution code is kept on the obligation type.

**Distribution Code for Calculated Adjustments.** Depending on the algorithm used for the [calculated adjustment](#), the distribution code may come from the adjustment type or the calculation lines of the algorithm. If the adjustment's calculation algorithm gets the distribution code from the calculation lines, you do not need to specify a distribution code on the adjustment type.

For more information about the source of the distribution codes on financial transactions, see [The Source Of GL Accounts On Financial Transactions](#).

Use **Adjustment Type Category** to assign a broad category to the adjustment type. Valid values are **Assessment**, **Credit Penalty & Interest**, **Waiver**, **Write Off**.

**Note.** The values for this field are customizable using the Lookup table. This field name is ADJ\_TYPE\_CAT\_FLG.

Define the **Debt Category** to assign to debit adjustments created for this adjustment type.

**Required for base algorithms.** The base P&I calculation algorithm and the base [Determine Detailed Balance](#) algorithm require that every debit adjustment reference a debt category.

If this adjustment type is for a certain type of credit adjustment and it has special rules for how credits are applied in the base Determine Detailed Balance algorithm, define the appropriate **Debt Category Priority**. Refer to [Debt Categories and their Priorities](#) for more information.

Enter the **Currency Code** for adjustments of this type.

Turn on **Sync. Current Amount** if adjustments of this type exist to force an obligation's current balance to equal its payoff balance. These types of adjustments are issued before an obligation's funds are transferred to a write-off obligation. If this switch is on, choose an **Adjustment Fin Algorithm** that does not impact payoff balance or the GL, but does affect the obligation's current balance (refer to [ADJT-CA](#) for an example of such an algorithm).

Enter a **Default Amount** if an amount should be [defaulted](#) onto adjustments of this type.

For more information about current and payoff amounts, refer to [Current Amount versus Payoff Amount](#).

If the AP Adjustment should be recorded in respect of the taxpayer's income statement amounts, indicate the **Income Statement**. The values of this field are **Interest** and **Miscellaneous**. This type of adjustment would also have an **A/P Request Type Code** selected, as income statement reporting is handled in A/P.

Turn on **Print By Default** if information about adjustments of this type should print on the account's next bill.

Choose an **A/P Request Type Code** if this adjustment is interfaced to accounts payable (i.e., it's used to send a refund check to a taxpayer). Refer to [A/P Check Request](#) for more information.

The **Adjustment Freeze Option** defines when adjustments can be frozen and therefore when an obligation's balance and the general ledger are affected by an adjustment. Refer to [Preventing Obligation Balances And The GL From Being Impacted Until Bill Completion](#) to understand the significance of this option. Also note, if the [installation option's](#) Bill Segment Freeze Option is **Freeze At Will**, this field is defaulted to **Freeze At Will** and cannot be changed.

**Warning!** Adjustment types for adjustments created during bill completion (e.g., by a bill completion algorithm) must have their adjustment freeze option set to **Freeze At Will**. Otherwise (i.e., if the option is **Freeze At Bill Completion**) they will not be frozen until a subsequent bill is completed.

If adjustments of this type require approval, define an **Approval Profile**. For more information, refer to [The Big Picture of Adjustment Approvals](#).

Enter the verbiage to appear on the printed bill in **Description on Bill**.

Use an **Adjustment Business Object** to define a [BO](#) that may govern additional rules related to adjustments of this type.

Use the characteristics collection to define a **Characteristic Type** and **Characteristic Value** common to all adjustments of this type. These can be used for reporting purposes or in your algorithms.

#### Adjustment Type - Adjustment Characteristics

To define characteristics that can be defined for adjustments of a given type, open **Admin Menu**, **Adjustment Type** and navigate to the **Adjustment Characteristics** tab.

#### Description of Page

Use the **Adjustment Characteristics** collection to define characteristics that can be defined for adjustments of a given type. Turn on the **Required** switch if the **Characteristic Type** must be defined on adjustments of a given type. Enter a **Characteristic Value** to use as the default for a given **Characteristic Type** when the **Default** switch is turned on. Use **Sequence** to control the order in which characteristics are defaulted.

#### Adjustment Type - Algorithms

#### Description of Page

The grid contains **Algorithms** that control important adjustment functions. If you haven't already done so, you must [set up the appropriate algorithms](#) in your system. You must define the following for each algorithm:

- Specify the **System Event** with which the algorithm is associated (see the table that follows for a description of all possible events).
- Specify the **Sequence Number** and **Algorithm** for each system event. You can set the **Sequence Number** to 10 unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event**.

System	Optional /	Description
--------	------------	-------------

Event	Required	
<i>Adjustment Cancellation</i>	Optional	<p>When an adjustment is canceled an algorithm of this type may be called to do additional work.</p> <p>Refer to <a href="#">The Lifecycle Of An Adjustment</a> for more information about canceling an adjustment.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Adjustment Freeze</i>	Optional	<p>When an adjustment is frozen an algorithm of this type may be called to do additional work.</p> <p>Refer to <a href="#">The Lifecycle Of An Adjustment</a> for more information about freezing an adjustment.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Adjustment Information</i>	Optional	<p>We use the term "Adjustment information" to describe the basic information that appears throughout the system to describe an adjustment. The data that appears in "Adjustment information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the "Adjustment information" algorithm on installation options or the system default "Adjustment information" if no such algorithm is defined on installation options.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Adj. Financial Transaction</i>	Required	<p>Algorithms of this type are used to construct the actual financial transaction associated with the adjustment. The financial transaction controls the adjustment's affect on the obligation's payoff and current balances. It also constructs the information that is eventually interfaced to your general ledger.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Default Adjustment Amount</i>	Optional	<p>Algorithms of this type are used to default the adjustment amount. Refer to <a href="#">Default the Adjustment Amount</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Determine Obligation</i>	Optional	<p>Algorithms of this type are used to find an obligation for which the adjustment can be posted. This plug-in is used particularly during adjustment upload when a staging record does not identify the obligation ID. Refer to <a href="#">Interfacing Adjustments From External Sources</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Resolve Suspense</i>	Optional	<p>Algorithms of this type are used to automatically resolve adjustments that are in suspense. Refer to <a href="#">Suspense Adjustments</a> for more information</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Generate Adjustment</i>	Optional	<p>Algorithms of this type are used to calculate the adjustment amount if an adjustment type indicates that the adjustment amount is calculated. Refer to <a href="#">Setting Up Calculated Adjustment Types</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Validate Adjustment</i>	Optional	<p>Algorithms of this type are used to validate information for the adjustment after it is generated.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>



## Setting Up Calculated Adjustment Types

You can use an algorithm to calculate an adjustment amount. Because the base package algorithm calculates adjustment amounts by calling the rate application, calculated adjustments are sometimes referred to as ratable adjustments.

**Ratable Adjustments Appear Deceptively Simple.** But, they are not. Calculated adjustments that use the base package algorithm have all the power and flexibility (and complexity) of the rate application. Anything that you can do with a rate can be applied to a calculated adjustment.

Adjustment types that indicate they are calculated have a generate adjustment algorithm. The base package algorithm defines the rate schedule used to calculate the adjustment as well as any UOM, TOU or RQI parameters.

To set up calculated adjustment types using the base package generate adjustment algorithm type:

- Define the rate that performs the calculations, including the rate schedule, rate version and rate components. Refer to [How To Create A New Rate](#) for information.

**Note.** If you create your own Generate Adjustment algorithm type, you may not need to set up a rate that performs the calculations. It depends on the needs of your algorithm type.

- Create a Generate Adjustment algorithm (refer to [Setting Up Algorithms](#)) that references the base package algorithm type that generates calculated adjustments (see the table above).
- If you want the generation algorithm's calculation lines to provide the distribution codes when the adjustment is posted to the GL, create an Adjustment Financial Transaction algorithm (refer to [Setting Up Algorithms](#)) that references an algorithm type that creates the adjustment's financial transactions using the calculation lines. A parameter of the adjustment financial transaction algorithm determines whether the distribution codes are taken from the adjustment type (AT) or calculation lines (CL). The system comes supplied with several sample algorithm types that [create adjustment financial transactions](#).
- Create an adjustment type where the **Adjustment Amount Type** is **Calculated Amount**, the **Generate Adjustment** event references the generation algorithm created above, and the **Adj. Financial Transaction** event references the adjustment financial transaction algorithm created above.

## Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_ADJ\\_TYPE](#).

## Setting Up Adjustment Type Profiles

Adjustment type profiles categorize your adjustment types into logical groups. When you link a profile to an obligation type, you limit the type of adjustments to be linked to the obligation type's obligations. The creation of adjustment profiles and their linkage to obligation types prevents inappropriate adjustments from being linked to your obligations. More than one adjustment type profile may be linked to an obligation type.



For example, you can create an adjustment type profile called *Miscellaneous Fees* and link to it the miscellaneous fee adjustment types. Then, you would link this profile to those obligation types that are allowed to levy such fees.

**Bottom line.** An adjustment can only be linked to an obligation if its adjustment type is part of an adjustment type profile that is valid for the obligation's obligation type. If an adjustment type is not linked to a profile, it could never be levied.

To set up adjustment type profiles, open **Admin Menu, Adjustment Type Profile**.

### Description of Page

Enter a unique **Adjustment Type Profile** and **Description** for the adjustment type profile.

Indicate the **Adjustment Types** that are part of the profile.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_ADJ\\_TYP\\_PROF](#).

## The Big Picture Of Adjustment Approval

Some implementations require adjustments to be approved by one or more managers before they impact an obligation's balance and the general ledger. For example, an adjustment used to refund a credit balance may require managerial approval before the refund is sent to the taxpayer. The topics in this section describe how to set up the system to support the approval of adjustments.

### Contents

- [Approval Is Controlled By Approval Profiles](#)
- [Approval Profiles Can Be Linked To Multiple Adjustment Types](#)
- [Adjustments Created In Batch Are Not Approved](#)
- [Approval Inserts A Step Into An Adjustment's Lifecycle](#)
- [Approval Requests Manage And Audit The Approval Process](#)
- [To Do Entries Are Created To Notify Approvers](#)
- [Monitoring and Escalating Approval Requests](#)
- [Rejecting Deletes The Adjustment](#)
- [Designing Your Approval Profiles](#)
- [Exploring Adjustment Approval Data Relationships](#)
- [Implementing Other Approval Paradigms](#)

### Approval Is Controlled By Approval Profiles

An approval profile contains the rules that define if and how an adjustment is approved. If an adjustment type does not reference an approval profile, the related adjustments do not require third-party approval before they impact an obligation's debt. If an adjustment type references an approval profile, the approval profile's approval hierarchy defines if the adjustment requires approval and who the authorized approvers are. For example, an approval profile can be configured with the following approval hierarchy:

- Adjustments < \$0 require approval by the "credit approvers role"
- Adjustments >= \$0 and <= \$10 do not require approval

- Adjustments > \$10 and <= \$100 require the approval of a user that belongs to the "level 1 approvers role"
- Adjustments > \$100 require two levels of approval: first a user that belongs to the "level 1 approvers role" must approve the adjustment; afterwards, the adjustment must be approved by a user that belongs to the "level 2 approvers role"

**Transfer adjustments.** The term "transfer adjustment" refers to two adjustments that are used to transfer moneys between two obligations. The adjustment with the positive amount is considered to be the debit adjustment; the other adjustment is considered the credit adjustment. When a transfer adjustment requires approval, only one of the adjustments needs to be approved. You control whether the debit side or the credit side of a transfer adjustment is used to control the approval process when you set up the approval profile.

### Approval Profiles Can Be Linked To Multiple Adjustment Types

Approval hierarchies are frequently the same for many adjustment types. The system allows an approval profile to be linked to multiple adjustment types to simplify the definition and maintenance of the rules over time.

### Adjustments Created In Batch Are Not Approved

The system assumes that no approval is necessary for adjustments created by batch processes even those whose adjustment type references an approval profile.

### Approval Inserts A Step Into An Adjustment's Lifecycle

[The Lifecycle Of An Adjustment](#) explains how an adjustment is transitioned from the **Freezable** state to the **Frozen** state when it should impact the general ledger and the obligation's balance. If an adjustment's adjustment type references an approval profile, the user cannot freeze the adjustment directly. Rather, the user must submit the adjustment for approval when it's ready and only when the last applicable approver approves the adjustment will it become **Frozen**.

**Freeze during bill completion.** You can configure the system to only freeze certain types of adjustments when the next bill is completed for the adjustment's account. When the last approver approves such adjustments, they remain in the **Freezable**. When the next bill is completed for the account, these adjustment become **Frozen**. Such adjustments that have not been approved at the time of bill completion will remain in the **Freezable** state. Refer to [Preventing Obligation Balances And The GL From Being Impacted Until Bill Completion](#) for more information.

### Approval Requests Manage And Audit The Approval Process

Users submit an adjustment for approval using a dedicated button on the [Adjustment](#) page. When an adjustment is submitted for approval, the system creates an "approval request". The approval request determines if the adjustment requires approval and, if so, the list of approvers. If the adjustment does not require approval, the approval request is updated to indicate such and the adjustment is **Frozen** immediately (if freezing is allowed prior to bill completion). If the adjustment requires approval, the approval request's state becomes **Approval In Progress** and the approver(s) are notified.

**Approval submission logic is customizable.** The previous paragraph describes how the base-package works when an adjustment is submitted for approval. This logic resides in an algorithm that's plugged in on the **C1-AdjustmentApprovalProfile** business object in the **Determine Approval Requirements** system event. Your implementation can change this logic by developing a new algorithm and plugging it into this business object. If your logic is meant to supersede the base-package algorithm, remember to inactivate the base-package algorithm by adding an appropriate inactivation option to this business object.

### To Do Entries Are Created To Notify Approvers

When an approval request detects an adjustment requires approval, it notifies the first approver by creating a To Do entry. The To Do entry is created using the To Do type and To Do role defined on the approval profile. All users who belong to the approving To Do role can see the entry. When a user drills down on an adjustment approval To Do entry, the [Adjustments - Approval](#) portal is opened. This portal contains summary information about the adjustment and the approval history of the adjustment. This portal is also where the user approves or rejects the adjustment.

When the first user in the To Do role approves an adjustment, the To Do entry is **Completed** and the approval request's audit log is updated. If there are no higher levels of approval required, the adjustment is **Frozen** (if freezing is allowed prior to bill completion) and the approval request is moved to the **Approved** state. If there are higher levels of approval required, a new To Do entry is created to the next To Do role in the approval hierarchy.

**To Do entries can create email.** A To Do entry can be configured to create an email message for every user in the To Do role to inform the user(s) of new adjustments requiring their attention. Refer to [To Do Entries May Be Routed Out Of The System](#) for the details.

### Monitoring and Escalating Approval Requests

The base-package is supplied with an algorithm that your implementation can use to monitor approval requests that have been waiting too long for approval. This algorithm can complete the current To Do entry and create a new one for a different role when the timeout threshold defined on the algorithm's parameters is exceeded. If you've configured the system to send email for approval, this algorithm can also send x reminder emails (where x is defined on the algorithm's parameters) before the approval request is escalated to the new To Do role. Refer to [C1-APR-TMOUT](#) for more information about this algorithm. If you plan to enable this functionality, plug-in your configured algorithm on the **Approval In Progress** state on the **C1-AdjustmentApprovalRequest** business object.

### Rejecting Deletes The Adjustment

When an adjustment is being approved, anyone with access to the adjustment can reject it by using the [Adjustments - Approval](#) portal. Users other than the current approver are allowed to reject an adjustment to allow an "in process" an adjustment to be withdrawn.

When an adjustment is rejected, the following takes place:

- The user is prompted for a reject reason.
- The approval request's audit log is updated with the reject reason and the approval request is moved to the **Rejected** state.
- The adjustment is deleted.

## Designing Your Approval Profiles

The following points describe a recommended design process:

- Create logical groups of adjustment types where each group has the same monetary hierarchy and approvers. An approval profile will be required for each of these groups.
- The number of To Do types (if any) that need to be created is dependent on how the adjustment approval To Do entries should be organized on To Do lists. For example, if all approval request To Do entries can appear in the same To Do list, you can use the base-package adjustment approval To Do type. However, if your organization prefers each approval profile's To Do entries to appear in a distinct To Do list, a separate To Do type will be needed for each list. Note that the base-package is supplied with a To Do type called [C1-ADAPP](#) that should be used as the basis for any new approval request To Do type.
- The number of To Do roles is dependent on who approves your adjustments. At a minimum, you will require a separate To Do role for each level in your approval profiles. Remember that every user in a To Do role will see its entries (and receive email if you've configured the system to do such).
- Refer to [Monitoring and Escalating Approval Requests](#) for how to configure the system to escalate approval requests that have been waiting too long.
- If your implementation requires email notification when an adjustment requires approval, the following setup is required:
  - Set up an outbound message type, external system, and XAI sender. Refer to [To Do Entries May Be Routed Out Of The System](#) for the details.
  - Every To Do type referenced on your approval profiles should be configured as follows:
    - Define the [F1-TDEER](#) batch process as the To Do type's routing process
    - Set up an algorithm that references the [C1-ADJAREQEM](#) algorithm type and plug it in the External Routing system event.

## Exploring Adjustment Approval Data Relationships

Use the following links to open the application viewer where you can explore the physical tables and data relationships behind the approval functionality:

- Click [C1-APPR PROF](#) to view the approval profile maintenance object's tables.
- Click [C1-APPR REQ](#) to view the approval request maintenance object's tables.

## Implementing Other Approval Paradigms

The above sections describe how the base-package adjustment approval process works. Because adjustment approval has been implemented using the **C1-AdjustmentApprovalProfile** and the **C1-AdjustmentApprovalRequest** business objects, your implementation can add additional business rules and change the approval user interface as required. Alternatively, if your implementation has a radically different approval process, you can create different business objects with their own business rules.

## Setting Up Approval Profiles

Approval profiles contain the rules that control how adjustments are approved. To set up an approval profile, open **Admin Menu, Approval Profile**.

Refer to [The Big Picture Of Adjustment Approval](#) for a detailed description of how approval profiles govern the adjustment approval process.

The topics in this section describe the base-package zones that appear on the Approval Profile portal.

### Contents

- [Approval Profile List](#)
- [Approval Profile Actions](#)
- [Approval Profile](#)
- [Approval Profile's Adjustment Types](#)

### Approval Profile List

The Approval Profile [List zone](#) lists every approval profile. The following functions are available:

- Click the [broadcast](#) icon to open other zones that contain more information about the adjacent approval profile.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each approval profile.

Click the **Add** link in the zone's title bar to add a new approval profile.

### Approval Profile Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions are available.

### Approval Profile

The Approval Profile zone contains display-only information about an approval profile. This zone appears when an approval profile has been broadcast from the Approval Profile List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Approval Profile's Adjustment Types

The Approval Profile's Adjustment Types zone lists every [adjustment type](#) that is governed by this approval profile. This zone appears when there is at least one adjustment type governed by the approval profile displayed in the Approval Profile zone.

## Tender Management

When a payment is received, a tender is created to record what was remitted (e.g., cash, check, credit card). The topics in this section describe control tables that must be set up in order to remit tenders.

We strongly recommend [Tender Management and Workstation Cashiering](#) before setting up the control tables described in this section.

### Contents

- [Setting Up Tender Types](#)
- [Setting Up Tender Sources](#)

## Setting Up Tender Types

Tender types are used to indicate the method in which the tender was made. A unique **Tender Type** must exist for every type of tender that can be remitted. For example, if you allow cash, checks, direct debits from a checking account, and direct debits from a credit card to be tendered, you'd need the following tender types:

Tender Type	Description	Like Cash	Generate Auto Pay	Require External Source ID	Require Expiration Date	External Type
CASH	Cash	Yes	No	N/A	N/A	N/A
CHEC	Check	No	No	N/A	N/A	N/A
OVUN	Cash drawer – over/under	No	No	N/A	N/A	N/A
DDCH	Direct debit - checking	No	Yes	Yes	No	<i>Checking withdrawal</i>
CRED	Direct debit – credit card	No	Yes	No	Yes	<i>Credit card withdrawal</i>

Go to **Admin Menu, Tender Type** to define your tender types.

### Description of Page

Enter a unique **Tender Type** and **Description** for the tender type.

Turn on the **Like Cash** switch if this tender type is cash or the equivalent of cash. This indicator controls if the system generates a warning if a cash-only account remits a tender other than cash. It is also used to generate a warning for online cashiers to turn in their tenders when the cash-like amount exceeds the maximum amount balance defined for the [tender source](#).

Turn on **Generate Auto Pay** if this type of tender causes an automatic payment request to be routed to a financial institution. For example, this switch will be on if this tender type is used for direct debits from a taxpayer's checking account (because every tender of this type will have an automatic payment request created when the tender is created).

The following fields are only used for tender types associated with automatic payments:

#### External Type

This field is used by the background process that creates the information that is interfaced to the automatic payment source. Specifically, it controls the record type associated with the different types of automatic payments that are routed to the automated clearinghouse (ACH).

**Note.** The values for this field are customizable using the Lookup table. This field name is EXT\_TYPE\_FLG.

**Require Ext. Src. ID**

This switch indicates if an Auto-Pay Source that references this type of tender must contain an External Source ID. The External Source ID is the unique identifier of the financial institution to which the automatic payment will be routed.

This switch is typically turned on for tender types associated with checking / saving direct debits. It is turned off for tender types associated with credit card debits (you don't need an external source for a credit card debit, you just need the credit card number).

**Expiration Date Required**

Turn this switch on if an Auto-Pay Option that references an auto-pay source that references this type of tender must also contain an expiration date (e.g., automatic debit / credit cards).

Turn this switch off for tender types associated with checking / saving direct debits.

Turn on **Allow Cash Back** if the system should automatically calculate a cash back amount when a tender is remitted for this tender type and the amount tender exceeds the amount being paid.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_TENDER\\_TYPE](#).

## Setting Up Tender Sources

A unique **Tender Source** must exist for every potential source of funds. For example,

- Every cashiering station will have a unique tender source.
- Every lock box will have a unique tender source.
- Your remittance processor will have a unique tender source.
- If you allow taxpayers to pay bills automatically (e.g., via EFT), you'll need a tender source for each institution to which you route automatic payment requests. For example, if you route automatic payment requests to the automated clearinghouse (ACH), you'll need a tender source for the ACH.

For example, if you have 3 lock boxes, 2 cash drawers at an area office A, 2 cash drawers at area office B, and a single remittance processor, you'd need the following tender sources:

Tender Source	Type	External Source ID (Lockbox ID)	Default Starting Balance	Currency Code	Suspense Obligation
CASH-A01	Cashiering	N/A	150.00	USD	N/A
CASH-A02	Cashiering	N/A	150.00	USD	N/A
CASH-B01	Cashiering	N/A	150.00	USD	N/A



CASH-B02	Cashiering	N/A	150.00	USD	N/A
LB-INDUS	Lockbox	112910-A	N/A	USD	9291019281
LB-COMM	Lockbox	938219-C	N/A	USD	4739837372
LB-RESID	Lockbox	372829-B	N/A	USD	1912910192
REMIT	Lockbox	N/A	N/A	USD	1920038437
ACH	Auto Pay	N/A	N/A	USD	N/A

To set up a tender source, select **Admin Menu, Tender Source**.

### Description of Page

Enter an easily recognizable **Tender Source** and **Description** for the tender source.

Define the **Tender Source Type**. Valid values are: **Ad Hoc, Auto Pay, Online Cashiering** and **Lockbox**. The system uses this information to prevent tender controls from different sources from being included under the same deposit control. In other words, you can't mix ad hoc, automatic payment, cashiering and lockbox tenders under the same deposit control.

For more information, refer to [Maintaining Deposit Controls](#).

If the source is an external system (e.g., a lockbox or an automatic payment destination), use **External Source ID** to define the unique identifier of the source. The background process that interfaces tenders from this source uses this information to create the appropriate tender control when it interfaces payments from external sources.

If this source is a cash drawer, define the **Default Starting Balance**. This balance is defaulted onto new tender controls and may be overridden.

**Note.** The tender type of the **Start Balance** is defined on the installation record.

If this source is a cash drawer, define the **Max Amount Balance**. When the amount of **cash-like** tenders in a cash drawer exceeds this balance, a warning is issued to remind the cashier to turn in some of the funds to a tender control.

Define the **Currency Code** of tenders linked to this source. All tenders in a source must be of the same currency.

If this tender source is associated with payments that are **interfaced from an external source** (e.g., a lockbox or a remittance processor), use Suspense **Obligation** to define the obligation whose account will hold uploaded payments with an invalid account. Refer to [Payment Upload Error Obligations](#) for more information about suspense obligations. Also note, because the payment upload process simply books payments that reference invalid accounts to the account associated with this obligation, this account should belong to an account type with the appropriate payment distribution algorithms. This may entail creating a new account type that will only be used on these "suspense accounts". This account type would need the following algorithms:

- We'd recommend using a simple payment distribution algorithm like [PYDIST-PPRTY](#) (distribute payment based on obligation type's payment priority).
- We'd recommend using an overpayment distribution algorithm like [OVRPY-PPRTY](#) (distribute overpayment to highest priority obligation type).



Define the **Bank Code** and **Bank Account** into which the tender source's moneys will be deposited. The bank account defines the distribution code used to build the GL details for the payment. Refer to The [Source of GL Accounts on Financial Transactions](#) for more information. Note that the bank code and bank account can later be overwritten when entering Tender Deposits on [Deposit Control](#).

If this tender source is associated with payments that are [interfaced from an external source](#), for example tender sources associated with **Auto Pay** and **Lockbox Tender Source Types**, the information is also used as follows:

- The [payment upload process](#) uses this information to populate the bank and bank account when it creates deposit control records for the tender controls it creates during the interface. Refer to [Managing Payments Interfaced From External Sources](#) for more information.
- The [automatic payment interface](#) uses this information to populate the bank and bank account when it creates deposit control records for the tender controls it creates during the interface.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_TNDR\\_SRCE](#).

## Automatic Payment Options

If your taxpayers can pay their bills automatically (via direct debit or credit card debits), you'll need to set up the various control tables described in this section.

**Important!** Besides the tables described in this section, additional values must also be added to control tables defined under [Tender Management](#). Specifically, refer to [Setting Up Tender Types](#) and [Setting Up Tender Sources](#).

Refer to [Automatic Payments](#) for more information about how automatic payments are handled in the system.

#### Contents

[Setting Up Auto-Pay Route Types](#)  
[Setting Up Auto-Pay Source Codes](#)

## Setting Up Auto-Pay Route Types

Auto Pay Route Types are used to control when and how automatic payment requests are routed to a financial institution, and when the general ledger is impacted. Select **Admin Menu, Auto Pay Route Type** to define your route types.

#### Description of Page

To modify an auto pay route type, simply move to a field and change its value.

To add a new route type, press + to insert a row, then fill in the information for each field. The following fields display:

<b>Route Type</b>	The unique identifier of the route type.
<b>Description</b>	The description of the route type.
<b>Tender Source</b>	<p>The background process that routes automatic payment requests to a financial institution (e.g., the automated clearing house interface) will mark each automatic payment's associated tender with a tender control for audit and control purposes. The following points describe how this happens:</p> <ul style="list-style-type: none"> <li>• When the system sees that it's time to send an automatic payment to a financial institution, it looks at the automatic payment's auto-pay source.</li> <li>• Every auto-pay source references an auto-pay route type.</li> <li>• Every auto-pay route type references a tender source.</li> <li>• A <b>Tender Source</b> has a tender control for each group of tenders deposited / interfaced together one batch.</li> <li>• The system marks each automatic payment's associated tender with the latest tender control for the <b>Tender Source</b>. The system will create a new tender control each time it routes automatic payments to the tender source. Refer to <a href="#">Managing Payments Interfaced From External Sources</a> for more information about tender source and tender control.</li> </ul>
<b>Extract Batch Cd</b>	This field defines the background process that interfaces the automatic payment requests to the financial institution.
<b>Autopay Date Calculation Alg</b>	<p>This algorithm populates 3 dates associated with the automatic payment: 1) the date the automatic payment will be sent to the financial institution, 2) the date the general ledger will be impacted by the automatic payment, 3) the date of the payment.</p> <p>If you haven't done so already, you must set up this algorithm in the system. To do this, create a new algorithm (refer to <a href="#">Setting Up Algorithms</a>). On this algorithm, reference an Algorithm Type that populates automatic payment dates. Click <a href="#">here</a> to see the algorithm types available for this plug-in spot.</p>
<b>Where Used</b>	<p>Follow this link to open the data dictionary where you can view the tables that reference <a href="#">CI APAY_RT_TYPE</a>.</p>

## Setting Up Auto-Pay Source Codes

A unique **Auto-Pay Source** must exist for every bank / credit card company / bill payment service that your taxpayer's use as the source of the funds when they sign up for automatic payment. For example,

- Every bank will have a unique auto-pay source.
- Every credit card company will have a unique auto-pay source.

To set up an auto-pay source, select **Admin Menu, Auto Pay Source Type**.

### Description of Page

Enter an easily recognizable **Auto Pay Source Code** and **Description** for the auto-pay source.

The **Source Name** is the name of the financial institution.

When the system creates an automatic payment request, it also creates an associated payment tender. This tender (like all tenders) must have a tender type. This field defines the **Tender Type** associated with this auto-pay source's tenders. Refer to [Setting Up Tender Types](#) for more information.

The **External Source ID** is the unique identifier of the financial institution to which the automatic payment will be routed (e.g., the bank routing ID of the bank). This field is typically blank on automatic payments routed to credit card companies because the credit card company doesn't have an external source ID (whereas direct debits from banks must have a bank routing number). Whether this field is required is controlled by the **Tender Type**.

The **Auto Pay Route Type** controls when and how automatic payment requests get routed to a financial institution. It also controls when the general ledger is impacted by the automatic payments financial transaction. Refer to [Setting Up Auto-Pay Route Types](#) for more information.

The **Work Calendar** defines the financial institution's workdays. This information is used to determine the date on which automatic payment requests will be sent to the financial institution. Refer to [Setting Up External Workday Calendars](#) for more information.

The **Validation Algorithm** defines how the system validates the taxpayer's account ID at the financial institution. If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that validates the taxpayer's account ID at the financial institution. Click [here](#) to see the algorithm types available for this plug-in spot.

Refer to [Account – Auto Pay](#) for more information.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI APAY\\_SRC](#).

## Payment Advices

The topics in this section provide background information about payment advice functionality.

**This section is only relevant for some organizations.** The system configuration requirements described in this section are only relevant if your organization issues payment advices to the taxpayer instead of initiating electronic funds transfer directly to the taxpayer's bank.

## Contents

[What Is A Payment Advice?](#)

[Payment Advice vs. Direct Debit](#)

[Setting Up The System To Enable Payment Advices](#)

## What Is A Payment Advice?

Payment advice is a money order that is established at the initiative of the tax authority. When a bill is completed, the tax authority sends the taxpayer a document that indicates a payment amount and the taxpayer's bank details. If the taxpayer agrees to the information on the payment advice, he or she signs it and returns it to the clearinghouse address that is indicated on the payment advice. The clearinghouse, in turn, sends the dated and signed payment advice to the taxpayer's bank, which completes the payment.

## Payment Advice vs. Direct Debit

The existing functionality that creates automatic payments is referred to as direct debit processing. Payment advice processing differs from direct debit processing in the way that automatic payments get initiated. With payment advice processing, the usual automatic payment records - i.e. payment event, payment, tender and auto pay clearinghouse staging - are not created. Instead, a payment advice is printed and sent to the taxpayer. The taxpayer sends the approved payment advice directly to the clearinghouse.

**Note.** The system does not provide sample processes for extracting and printing payment advice information. Your implementation team would have to create these.

## Setting Up The System To Enable Payment Advices

You must set up a [Feature Configuration](#) to define parameters that control payment advice functionality.

The following points describe the various **Option Types** that must be defined:

- ***Payment Advice Functionality Supported.*** This option controls whether the system allows for payment advice processing.
  - Enter **Y** if the system should allow for both direct debit and payment advice processing.
  - Enter **N** if the system should only allow for direct debit processing.
- ***Default Auto Pay Method.*** This option is used for defaulting the auto pay method on new account auto pay records.

Refer to [Account – Auto Pay](#) for more information on auto pay method.

**Note.** The system assumes direct debit processing if the above feature options are not defined.

## Payment Event Distribution

The base package, by default, creates a single payment for a payment event. If your business requires potentially many payments to be created when payment events are added, you'll need to set up the various control tables described in this section.

Refer to [Distributing A Payment Event](#) for more information about how payment event distribution is handled in the system.

### Contents

- [Making Payments Using Distribution Rules](#)
- [Common Distribution Rules](#)
- [Payments Linked To Assessments](#)
- [Setting Up The System To Use Distribution Rules](#)

## Making Payments Using Distribution Rules

As part of this method, one or more distribution details are provided at payment time along with the usual payment and tender information. Each distribution detail record references a distribution rule and a corresponding value. The distribution rule encapsulates the business rules that govern the distribution of the payment amount into payments using the specified value.

The type of value being captured on the distribution detail and the logic that uses it to create payments are defined on the [distribution rule](#).

### Contents

- [Rule Value](#)
- [Determine Tender Account](#)
- [Creating Payment\(s\)](#)
- [Rule Value Can Capture Additional Information](#)

### Rule Value

The primary use of the rule value is to identify the business entity whose balance is to be relieved by creating payment(s). In most cases where the payor account is the same as the payee account it may also be used to identify the tender account associated with the payment(s).

### Determine Tender Account

The very first step in processing a distribution detail is to identify the tender account (i.e. the payor) associated with the payment. To do that the system calls the **[Determine Tender Account algorithm](#)** defined on the distribution rule providing it with the rule value and other tender information.

### Creating Payment(s)

The business logic that distributes a payment amount into one or more payments(s) targeted towards the entity identified by a rule value is held in designated **[Create Payment algorithms](#)** defined on the distribution rule.

## Rule Value Can Capture Additional Information

A rule value can also be used to capture additional information provided at payment time, like address information, comments, etc. Obviously payment distribution details with this type of rule value should have a zero payment amount, as they are not real payments. These distribution details end up being linked to a payment event, but unlike other distribution details they do not contribute any payments. You can think of these details as payment event characteristics.

You don't have to set up a **Create Payment** algorithm for distribution rules intended solely to capture additional payment information.

The base package provides a sample Distribution Rule - Create Payment [C1-PAYFRM](#) that caters for multiple distributions rule values.

## Common Distribution Rules

Tax authorities have complex payment distribution rules. Most of these rules follow a similar pattern, with a few key differences.

The following lists common payment distribution methods:

- Pay a specific obligation.
- Pay all the accounts of the taxpayer.
- Pay an account.
- Pay a specific filing period. This is common for estimated payments where a payment form indicates the filing period. Some tax authorities may use paper forms in check processing that indicate the filing period that a payment should apply to. This information is keypunched with the payment. The base package provides Distribution Rule - Create Payment [C1-PAYFRM](#) for paying a filing period. Refer to [Pay A Form / Filing Period Algorithm](#) for more details.
- Pay a tax form. The payment is directed to the obligation on the tax form. The base package provides Distribution Rule - Create Payment [C1-PAYFRM](#) for paying a form. Refer to [Pay A Form / Filing Period Algorithm](#) for more details.
- Pay a pay plan. The payment is directed to the obligations that the pay plan covers. The base package provides a Distribution Rule - Create Payment algorithm [C1-DR-PAYPP](#) for paying a pay plan. Refer to [Distributing Payments for Pay Plans](#) for more details.
- Pay a collection case. The payment is directed to what the collection case is collecting on - e.g. obligations, assessments, etc.
- Pay a collection letter. A collection letter can be generated from overdue processing. The payment is directed to what the overdue process is collecting on - e.g. obligations, assessments, etc.

## Payments Linked To Assessments

When dynamic credit allocation is practiced, a payment targeted to a specific assessment must have the payment FT referencing the assessment's FT ID. If there is excess credit on the payment segment, it is dynamically allocated to any other assessments for the obligation.

The base package provides a payment freeze algorithm [C1-PYFZ-PYAS](#) that links a payment FT to a specific assessment. This algorithm looks for a match value on the payment that references the assessment. The assessment is linked to the payment by populating the assessment's FT ID on the payment FT's Group FT ID field.

For more information, refer to [Credit Allocation](#).

## Setting Up The System To Use Distribution Rules

### Contents

- [Setting Up Distribution Rules](#)
- [Feature Configuration](#)
- [Set Up Account Type](#)
- [Set Up Match Type](#)

### Setting Up Distribution Rules

Define a Distribution Rule for each payment event distribution method practiced by your business.

### Contents

- [Distribution Rule - Main](#)
- [Distribution Rule - Algorithms](#)

#### Distribution Rule - Main

To set up a distribution rule, navigate to **Admin Menu, Distribution Rule**.

#### Description of Page

Enter a unique **Distribution Rule** and **Description** for the distribution rule.

Provide a short and unique **Distribution Rule Label** to be used as rule's name throughout the system.

**Characteristic Type** defines the type of entity whose balance is relieved by the payment(s) this rule creates. For example, if this rule targets payments(s) towards a specific obligation, you'd reference a characteristic that its value identifies an obligation. We use the term "rule value" for the characteristic value.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference

[CI\\_DST\\_RULE](#)

#### Distribution Rule - Algorithms

Navigate to **Admin, Distribution Rule, Algorithms** to set up the algorithms appropriate for your distribution rule.

The **Algorithms** grid contains algorithms that control important functions. You must define the following for each algorithm:

- Specify the algorithm's **System Event** (see the following table for a description of all possible events).

- Specify the **Algorithm** to be executed when the System Event executes. Set the **Sequence** to **10** unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event** (note, all system event's are optional and you can define an unlimited number of algorithms for each event).

System Event	Optional / Required	Description
<i>Create Payment</i>	Optional	This algorithm is executed to distribute a payment distribution detail payment amount into one or more payments. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Determine Tender Account</i>	Optional	This algorithm is executed to determine the tender account associated with the payment distribution detail. Only one such algorithm may be specified. Click <a href="#">here</a> to see the algorithm types available for this system event.

## Feature Configuration


You must set up a [Feature Configuration](#) to define parameters that control various payment event distribution options. The following is an example of how the Feature Configuration would look for an implementation:

The following points describe the various **Option Types** that must be defined:

- Always Enable Distribution Rule.** This option controls whether the system should only use the distribution rule method to add payment events or rather allow both the default method and the distribution rule method to coexist.
  - Enter **Y** if the system should always use distribution rules. With this setting, navigation to the Payment Event page in add mode opens up the [Payment Event Quick Add](#) page (defaulting it to the **single** payment event dialog). This dialog is designed to create a payment event using distribution rules.
  - Enter **N** if the system should allow both methods. With this setting, navigation to the Payment Event page in add mode opens up the standard [Payment Event - Add Dialog](#) that uses the default method to create a payment event. If you want to use the distribution rule method, navigate to the Payment Event Quick Add page from the menu.
- Default Distribution Rule.** This option states your default distribution rule that appears throughout the system.

## Set Up Account Type

If you do dynamic credit allocation, set up the payment freeze algorithm [C1-PYFZ-PYAS](#).

 We recommend specifying **payment distribution** and **overpayment** algorithms because they are used as default logic in case distribution rules are not supplied.

## Set Up Match Type

If you use set the payment freeze algorithm [C1-PYFZ-PYAS](#), set up a match type for assessment. This match type should NOT reference an override payment distribution algorithm (if this algorithm is blank, the account type's payment distribution algorithm is used). Refer to [Payments Linked to Assessments](#) for more details.



## Cancel Reasons

---

As described in [The Financial Big Picture](#), the various types of financial transactions can be canceled if their financial impact needs to be reversed from the system. Whenever a financial transaction is canceled, a cancel reason must be specified. This section describes the control tables that contain the cancel reason codes.

### Contents

- [Setting Up Bill \(Segment\) Cancellation Reasons](#)
- [Setting Up Payment Cancellation Reasons](#)
- [Setting Up Adjustment Cancellation Reasons](#)

## Setting Up Bill (Segment) Cancellation Reasons

Open **Admin Menu, Bill Cancel Reason** to define your bill segment cancellation reason codes.

### Description of Page

Enter an easily recognizable **Bill Cancel Reason** and **Description** for the bill cancellation reason.

Only use **System Default** on those reason codes that are placed on bill segments that are automatically canceled by the system. Valid values are: **Turn off auto-cancel**, **Bad estimated read auto-cancel**, and **Mass Cancel**. The reason code identified as **Turn off auto-cancel** is placed on bill segments that are automatically canceled when the final bill segment ends before the prior bill (and therefore we have to cancel the prior bill). The reason code identified as **Bad estimated read auto-cancel** is placed on bill segments that are automatically canceled by the system when it detects that it used an estimated read whose consumption is greater than the next actual read (and therefore we have to cancel the estimated bill segment). The reason code identified as **Mass Cancel** is placed on bill segments that are canceled as a result of the execution of the Mass Cancellation background process. Refer to [Mass Cancellation](#) for more information.

**Required values.** You must have one reason code defined for each of the System Default values.

## Setting Up Payment Cancellation Reasons

Open **Admin Menu, Pay Cancel Reason** to define your payment cancellation reason codes.

### Description of Page

Enter an easily recognizable **Cancel Reason** and **Description** for the payment cancellation reason.

Turn on the **NSF Charge** switch if the system should invoke the non-sufficient funds (NSF) algorithm when a tender is cancelled using this reason code. Refer to [NSF Cancellations](#) for more information.

The next several fields are used to change an account's credit rating or cash-only points if a tender is canceled using the respective reason code.

- Use **Affect Cash-Only Score By** to define how tenders canceled using this reason will affect the account's cash-only score. This should be a positive number. When a taxpayer's cash-only points exceed the cash-only threshold amount defined on the CIS installation record, the account is flagged as cash only during payment processing and on Control Central.
- Use **Affect Credit Rating By** to define how tenders canceled using this reason will affect the account's credit rating. This should be a negative number. A taxpayer's credit rating is equal to the start credit rating amount defined on the CIS installation record plus the sum of credit rating demerits that are currently in effect.
- Use **Months Affecting Credit Rating** to define the length of time the demerit remains in effect. This information is used to define the effective period of the credit rating demerit record.

For more information, refer to [Account – Collections](#).

**The payor gets the credit rating / cash only hit.** When you cancel a tender you must specify a cancellation reason. If the cancellation reason indicates a credit rating / cash only demerit should be generated, the system levies the credit rating transaction on the PAYOR's account.

The **System Default Flag** is specified on those cancellation reasons that are placed on payment segments that are automatically cancelled by the system. Valid values are: **Re-opened Bill**. The **Re-opened Bill** value is used as follows:

- Payments are automatically created for accounts who pay their bills automatically when their bills are completed.
- If such a bill is reopened before the automatic payment is interfaced to the paying authority, the system automatically cancels the payment. The **Re-opened Bill** cancellation reason is placed on such payments.

## Setting Up Adjustment Cancellation Reasons

Open **Admin Menu, Adjustment Cancel Reason** to define your adjustment cancellation reason codes.

### Description of Page

Enter an easily recognizable **Cancel Reason** and **Description** for each adjustment cancellation reason.

## Miscellaneous Financial Controls

This section describes miscellaneous control tables.

### Contents

[A/P Check Request](#)  
[Bill Charge Line Type](#)

## A/P Check Request

Adjustments whose adjustment type is marked with an A/P check request code are interfaced to your A/P system. Your A/P system then cuts the checks.

Refer to [Controls The Interface To A/P and Income Statement Reporting](#) for more information about the accounts payable interface.

You must set up at least one A/P check request code if you want A/P to cut checks.

To set up A/P check request types, open **Admin Menu, A/P Request Type**.

### Description of Page

Enter an easily recognizable **A/P Request Type** for the accounts payable request type.

Use **Due Days** to define when the check is cut. The cut date is equal to the adjustment date plus due days.

Select a **Payment Method**. Choose from these options:

**System Check**

System check

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_APREQ\\_TYPE](#).

## Bill Charge Line Type

**Background information.** Before using this page, you should be comfortable with the topics described under [Setting Up Billable Charge Templates](#) and [Uploading Billable Charges](#).

Billable charge line types will simplify the effort required to interface billable charges from an external system. Each line type contains values that will be defaulted onto the line details associated with the uploaded billable charges. Obviously, this defaulting is possible only if you specify a billable charge line type on the billable charge upload staging lines.

To set up billable charge line types, select **Admin Menu, Bill Charge Line Type**.

### Description of Page

Enter an easily recognizable **Bill Charge Line External Type** and **Description**.

Use **Currency Code** to define the currency to be defaulted onto billable charge upload lines that reference this line type.

Use **Show on Bill** to define the value to be defaulted into the Show on Bill indicator on billable charge upload lines that reference this line type.

Use **App in Summary** to define the value to be defaulted into the App in Summary indicator on billable charge upload lines that reference this line type.

Use **Memo Only, No GL** to define the value to be defaulted into the Memo Only, No GL indicator on billable charge upload lines that reference this line type.

Use **Distribution Code** to define the values to be defaulted into the Distribution Code field on billable charge upload lines that reference this line type.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI BCHG UP XTYP](#).

## Payables Cash Accounting

In some cases certain amounts such as sales tax distributions and 3rd party liabilities are not truly payable until the taxpayer remits payment. We refer to this as "payables cash accounting". This practice should be contrasted with "payables accrual accounting" in which the liability is realized when the bill is created (as opposed to when it is paid).

If your organization does not practice payable cash accounting, you may skip this section as accrual accounting is the system default. If you practice payables cash accounting, the contents of this section describe how to configure the system appropriately.

### Contents

- [Accrual versus Cash Accounting Example](#)
- [Distribution Code Controls Cash Accounting For A GL Account](#)
- [Cash Accounting and Incurring Debt](#)
- [Cash Accounting and Relieving Debt](#)

## Accrual versus Cash Accounting Example

Accrual accounting means that all payables are booked when the debt financial transaction (bill segment or adjustment) is created.

If the payable is subject to payables cash accounting, the liability is not booked when the bill segment or adjustment is created, rather, the amount of the liability is placed in a "holding" GL account. When the taxpayer pays, the moneys are transferred from the "holding" GL account to the true tax payable account.

The following is an example of the financial events that transpire when a simple sales tax assessment is posted and a payment is received using accrual accounting.

Event	GL Accounting	County A Payable Balance
Tax assessment created	A/R 110	(10)
	Revenue - State <100>	
	Revenue - County A <10>	
Payment received	Cash 110 A/R <110>	(10)

In the above example, you'll notice that the payable is booked when the adjustment is created. Let's contrast this with what takes place if the payable is subject to payables cash accounting.

Event	GL Accounting	County A Payable	County A Holding
-------	---------------	------------------	------------------

		Balance	Balance
Tax assessment created	A/R 110	0	(10)
	Revenue - State <100>		
	County A Holding <10>		
Payment received	Cash 110	(10)	0
	A/R <110>		
	County A Holding 10		
	County A Payable <10>		

Notice that when the adjustment is created, the revenue for the county is not booked, rather, the amount is placed in a “holding” GL account. When the taxpayer pays, the moneys are transferred from the “holding” GL account to the true **County A** payable account.

If the above seems simple, consider the following complications that must be considered:

- What happens if a partial payment is received?
- What happens if there are multiple amounts subject to cash accounting rules?
- What happens if the payment is cancelled?
- What if the payment isn't received and we have to write-off debt?
- What happens if the taxpayer overpays?
- What happens if the obligation is subject to penalty and interest and dynamic credit allocation?

The above points, and more, are discussed below.

## Distribution Code Controls Cash Accounting For A GL Account

**Note.** If you do not understand the significance of distribution codes, please refer to [Setting Up Distribution Codes](#).

Whether or not cash accounting is used for a specific GL account is defined on HOLDING GL account's distribution code (i.e., the holding GL account references the true payable account).

It is very important that unique payable and holding distribution codes be used for each type of amount subject to cash accounting rules. For example, for sales tax distribution, a pair of payable and holding accounts is required for each separate distribution amount.

Without unique distribution codes for each payable and holding account, the system cannot keep track of how much of a given amount is being held, awaiting payment.

## Cash Accounting and Incurring Debt

When calculating debt that includes payables, the distribution codes associated with the payables (for example on the rate components that calculate payables) must be your HOLDING payable distribution codes. No other logic is needed for cash accounting at this time.

## Cash Accounting and Relieving Debt

The following section describes functionality required for obligations that practice cash accounting and do NOT practice penalty and interest or dynamic credit allocation.

**Dynamic Credit Allocation.** Refer to [P&I and Cash Accounting](#) for functionality required for obligations that do practice penalty and interest or dynamic credit allocation.

### Contents

[Payment Segment Financial Transaction Algorithms Transfer Holding Amounts to Payable GL](#)  
[How Does The System Know What Amounts To Transfer From Holding To Payables?](#)  
[Partial Payments Result In Partial Payables](#)  
[Adjustments That Behave Like Payments](#)  
[Overpayment Of Taxes Due To Cancel/Rebills](#)  
[Cash Refunds](#)  
[Over Payments](#)

## Payment Segment Financial Transaction Algorithms Transfer Holding Amounts to Payable GL Accounts

Logic exists in the pay segment's FT algorithm that transfers amounts from payable holding distribution codes to their respective payable real distribution codes.

Refer to [Setting Up Payment Segment Types](#) for how to define the appropriate FT algorithm.

The following table shows what happens to the financial transaction associated with the payment segment for a cash accounting taxpayer.

Event	GL Accounting
Adjustment is created	A/R 110 Revenue - State <100> County Holding <10>
Payment segment relieves receivables	Cash 110 A/R <110>
Additional GL details created when the payment segment FT algorithm transfers the holding amount to a payable account	County Holding 10 County Payable <10>
<i>Net affect of the above</i>	Cash 110 A/R <110>

	County Holding 10
	County Payable <10>

## How Does The System Know What Amounts To Transfer From Holding To Payables?

When a payment segment is created for an account that is subject to cash accounting processing, the system determines if there is a CREDIT balance for any holding distribution code in respect of the obligation. If so, it generates additional GL details to transfer moneys from the holding distribution code to the payable distribution code in proportion to the amount of receivables relieved by the payment. Therefore, if 100% of receivables are relieved by the payment segment, 100% of the holding amounts will be transferred to payable distribution codes. Refer to [Partial Payments Result In Partial Payables](#) for an example of what happens when a partial payment is created.

## Partial Payments Result In Partial Payables

The previous example showed the entire County holding amount being transferred to the County payable account. The entire holding amount was transferred because the obligation was paid in full. If a partial payment is received, only part of the holding amount will be transferred to the payable amount (proportional to the amount of receivables reduced by the payment). An example will help make the point.

Event	GL Accounting	County Payable Balance	County Holding Balance
Adjustment created	A/R 110 Revenue - State <100> County Holding <10>	0	(10)
Partial payment received	Cash 27.50 A/R <27.50> County Holding 2.50 County Payable <2.50>	(2.50)	(7.50)

## Adjustments That Behave Like Payments

There are several types of adjustments that behave just like payments (in respect of payables cash accounting), for example an overpayment transferred one obligation to another

The above events should cause the system to transfer holding amounts to true payable amounts (notice that the above examples are all transfer adjustments).

However, there are many other adjustments that should NOT behave like payments. You control how the adjustment works by selecting the appropriate FT algorithm when you [set up adjustment types](#) (refer to [ADJT-AC](#) and [ADJT-TC](#) for a description of the base package algorithms that causes the holding amounts to be manipulated in proportion to the amount of receivable being adjusted). In other words, there are adjustment FT algorithms that cause the transference of holding payable amounts to real payable amounts when the A/R balance is decreased by the adjustment.

**Cash refunds can behave like “negative payments”.** In addition to the above examples of transfer adjustments behaving like payments, you should be aware that cash refunds may impact your holding and true payable balances. Refer to [Cash Refunds](#) for more information.

### Overpayment Of Taxes Due To Cancel/Rebills

Lets assume a cancel / rebill occurs after a payment is received and the net affect of the cancel / rebill is that the taxpayer has overpaid their taxes.

This is an example of a tax that is "billed" where one obligation is used for ongoing billing of the tax amount.

Event	GL Accounting	County Payable Balance	County Holding Balance
Bill segment created	A/R 110 Revenue - State <100> County Holding <10>	0	(10)
Payment received	Cash 110 A/R <110> County Holding 10 County Payable <10>	(10)	0
Cancel	A/R <110> Revenue 100 County Holding 10	(10)	10
Rebill	A/R 27.50 Revenue <25> County Holding <2.50>	(10)	7.50

You'll notice that the amount payable to the county still indicates \$10 (the amount of amount that was paid by the taxpayer). However, you'll notice that the county holding balance is 7.50 (debit). This looks a bit odd, but it's correct. Remember that at this point, the taxpayer has a credit balance of \$75 and this will be whittled down as successive bills are produced as shown below. Note: refer to [Cash Refunds](#) for an example of what happens if you refund the credit with a check rather than letting it whittle down.

Event	GL Accounting	County Payable Balance	County Holding Balance
		(10)	7.50
Bill segment created	A/R 55 Revenue <50> County Holding <5>	(10)	2.50



Bill segment created	A/R 110	(10)	(7.50)
	Revenue <100>		
	County Holding <10>		

In the unlikely event of a payment being received while the county holding has a debit balance, nothing will be done in respect of transferring funds from holding to payable (there is nothing to transfer).

### Cash Refunds

If you refund moneys to a cash accounting taxpayer, it's important to do the opposite of what was done when the payment was received (i.e., you need to transfer the payable back to the holding account). The following example should help clarify this situation (this example shows a refund due to a credit balance that occurred as a result of a cancel/rebill).

Event	GL Accounting	County Payable Balance	County Holding Balance	Obligation's Payoff Balance
Bill segment created	A/R 110 Revenue <100> County Holding <10>	0	(10)	110
Payment received	Cash 110 A/R <110> County Holding 10 County Payable <10>	(10)	0	0
Cancel	A/R <110> Revenue 100 County Holding 10	(10)	10	(110)
Rebill	A/R 27.50 Revenue <25> County Holding <2.50>	(10)	7.50	(82.50)
Payment refunded (via an A/P adjustment)	Cash <82.50> A/R 82.50 County Holding <7.50> County Payable 7.50	(2.50)	0	0

We understand this is tricky, but consider this - when a cash accounting taxpayer makes a payment, the system transfers county holding CREDIT balances to county payable distribution codes in proportion to the amount of the receivable DEBIT amount that was reduced by the payment. Therefore, when cash is returned to the taxpayer, the system should transfer county holding DEBIT balances to county payable distribution codes in proportion to the amount of the receivable CREDIT that was reduced by the refund.

**Note.** The above takes place when an A/P adjustment is created if the related adjustment type references the appropriate FT algorithm (refer to [adjustment FT algorithm used for adjustments that behave like payments](#)).

## Over Payments

If a taxpayer overpays a tax amount (i.e., we receive more cash than receivables), we strongly recommend you set up the system to NOT keep the excess credit on the taxpayer's regular obligation. Rather, we recommend you segregate the receivable onto an "excess credit" obligation. If you do this, the system can transfer any excess credits to the regular obligations at bill completion time or at form posting time. When this transfer occurs, the same accounting described under [Payments Segment Financial Transaction Algorithms Transfer Holding Amounts To Payable GL Accounts](#) occurs as shown in the following example. Note: this example assumes an excess credit of \$110 was transferred to a normal obligation and the normal obligation had \$10 of held payables.

Refer to [Overpayment Obligations](#) for how to set up the system to segregate overpayments on a separate obligation.

**Why not keep excess credits on a taxpayer's regular obligation?** Because the system can't differentiate between a credit that exists as a result of an overpayment and a credit that exists because of cancel/rebills, it would be impossible for the system to know if payables should be realized as a result of the reduced credit balance. However, if you keep overpayments on an excess credit obligation, the system knows to treat any transference of these credits as "payments" and therefore it can transfer holding balances to true payables.

Event	Normal Obligation GL Accounting	Excess Credit Obligation GL Accounting
Bill segment created	A/R 110 Revenue <100> County Holding <10>	
Payment of \$300 is received	Cash 110 A/R <110> County Holding 10 County Payable <10>	Cash 190 Overpay <190>
Bill segment created	A/R 110 Revenue <100> County Holding <10>	
Transfer excess credit amount to normal obligation (when bill is completed).	Xfer 110 A/R <110>	Overpay 110 Xfer <110>
Because the transfer adjustment is the equivalent of a cash relief outstanding county holding is relieved in proportion to	County Holding 10 County Payable <10>	

the amount of receivables that are reduced by the transfer		
<i>Net affect of the transfer</i>	Xfer 110 A/R <110> County Holding 10 County Payable <10>	Overpay 110 Xfer <110>

## Open Item Accounting

The topics in this section provide background information about open-item accounting.

**This section is only relevant for some organizations.** The system configuration requirements described in this section are only relevant if your organization practices open-item accounting. If your organization practices balance-forward accounting, you need only indicate such on your [account types](#); no other setup is required. Refer to [Open Item Versus Balance Forward Accounting](#) for more information about these two accounting practices.

### Contents

- [Open-Item Versus Balance-Forward Accounting](#)
- [Accounting Method Defined On Your Account Types](#)
- [Match Events](#)
- [Disputing Items](#)
- [Overpayments](#)
- [Setting Up The System To Enable Open Item Accounting](#)
- [Setting Up Match Types](#)
- [Setting Up Match Event Cancellation Reasons](#)

## Open-Item Versus Balance-Forward Accounting

If you practice open-item accounting, you match payments against bills. The term "open-item accounting" is used to describe this accounting practice because:

- Payments are matched against "open items" (i.e., unpaid bills and adjustments)
- Only unmatched bills and adjustments (i.e., open items) affect aged debt.

Contrast open-item accounting with "balance-forward" accounting - in a balance-forward world, payments are not matched to bills. Rather, payments implicitly relieve a taxpayer's oldest debt.

## Accounting Method Defined On Your Account Types

You define the type of accounting method that is practiced ([balance-forward versus open-item](#)) on your [account types](#). The account type should be configured based on the accounting method practiced for that tax type.

## Match Events

Match events are used to match open-items (i.e., debit and credit financial transactions) together. The topics in this section provide an overview of match events.

## Contents

[Match Events Match Debit FTs To Credit FTs](#)  
[When Are Match Events Created?](#)  
[Match Event Lifecycle](#)  
[Payments And Match Events](#)  
[How Are Match Events Cancelled?](#)  
[Current Amount Is Matched, Not Payoff](#)

## Match Events Match Debit FTs To Credit FTs

For open-item taxpayers, the system matches credit financial transactions (FT's) to debit FT's under a [match event](#). The following are important points about match events.

- A match event may contain an unlimited number of FT's. For example, the match event can match 2 credit FTs against a single debit FT.
- A match event contains FTs associated with a single account. While the FTs under a match event may belong to multiple obligations, all FTs under a match event must belong to the same account.
- The status of the match event is **balanced** when the sum of the debits equals the sum of the credits. If debits do not equal credits, the status of the match event is **open** and the various FTs would still affect the taxpayer's aged debt. Refer to [Match Event Lifecycle](#) for more information.

## When Are Match Events Created?

The following points describe when match events are created for open-item accounts:

**Note.** Match events are only created for open-item accounts (i.e., those accounts with an account type that indicates open-item accounting is practiced). Match events may not be created for balance-forward accounts.

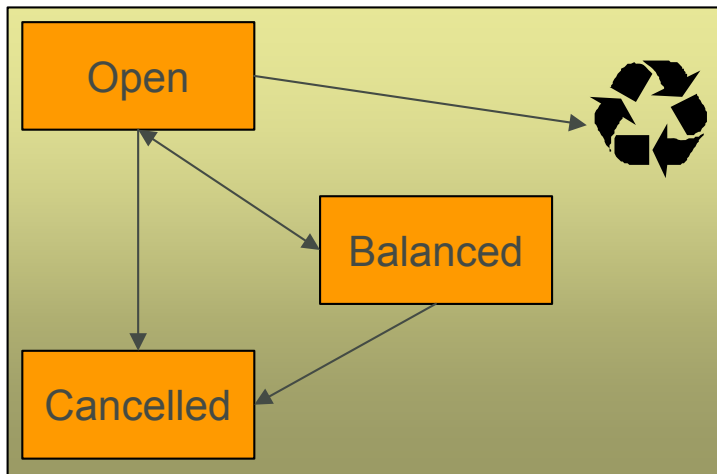
- The system can create one or many match events when a payment is added. This match event matches the payment's credit FTs with the debit and credit FTs from bill segments and adjustments. The FTs that are linked to the match event are controlled by the payment's **match type** and **match value** (payments made by open-item taxpayers must reference a match type and match value). Refer to [Payments And Match Events](#) for more information.
- The system may create a match event when any type of financial transaction is cancelled. This match event groups together the original FT with its cancellation FT. Refer to [How Are Match Events Cancelled?](#) for more information.
- The system creates a match event when a bill is completed for taxpayers that pay automatically (i.e., direct debit taxpayers). The match event groups together the bill's new charges against the automatic payment's payment segments.
- The system creates a match event when a bill is completed where the new charges are offset by other financial transactions.

Refer to [Bill Lifecycle](#) for more information about what happens during bill completion.

- The system creates a match event when an obligation closes and the obligation has unmatched FTs. For example, consider an obligation for debt that is written off. This obligation closes when the system creates transfer adjustments to transfer the debt to a write-off obligation (or writes down the debt). The system creates a match event to match the original debt to the transfer adjustments used to write-off the debt.
- A user can create a match event manually at any time. Manual match events would be created under a variety of situations. For example:
  - If a taxpayer disputes a charge. Refer to [Disputing Items](#) for more information about disputes.
  - To handle unusual situations when the system is unable to automatically match FT's together.

### Match Event Lifecycle

The following diagram shows the possible lifecycle of a match event:



Match events are initially created in the **open** state. Financial transactions (FT's) linked to **open** match events affect arrears, but not in an open-item fashion. Rather, FT's linked to **open** match events affect arrears in a balance-forward fashion. Refer to [Open Item Versus Balance Forward Accounting](#) for more information about these two accounting methods.

A user may delete an **open** match event. When an **open** match event is deleted, its FT's may be linked to other match events.

The system automatically changes an **open** event's status to **balanced** when the sum of the debit financial transactions (FT's) equals the sum of the credit FT's for each obligation on the match event. It's worth stressing that a match event may contain FT's from many obligation's and each obligation's FT's must sum to zero before the match event can become **balanced**.

A user may re**open** a **balanced** event (by adding / removing FT's so that the match event becomes unbalanced).

A user may **cancel** a **balanced** or **open** match event. Refer to [How Are Match Events Cancelled?](#) for more information about cancellation.

## Payments And Match Events

As described under [When Are Match Events Created?](#), the system creates a match event when a payment is added for an open-item account. The system uses the payment's **match type** and **match value** to determine the FT's (e.g., bill segments and adjustments) that will be matched with the payment's FT's (i.e., the payment segments).

Another way to think of this is as follows:

- When most payments are distributed, the system calls the payment distribution algorithm that is plugged-in on the account's account type.
- However, a payment that is made in respect of a specific bill requires a different distribution algorithm because the payment should only be distributed amongst the debt associated with the specific bill being paid. This is accomplished by referencing a match type / match value on the payment. The match type references the appropriate payment distribution algorithm. This algorithm is used rather than the account type distribution algorithm.

For example, if a payment were made in respect of bill ID **192910192101**, this payment would reference a match type of **bill ID** and a match value of **192910192101**. At payment distribution time, the system calls the override payment distribution algorithm associated with this match type. The base package bill ID distribution algorithm does several things:

- It distributes the payment amongst obligations associated with the bill.
- It creates a match event and links the bill's bill segment and adjustment FT's to it.
- Refer to the [Bill ID Match Type Algorithm](#) for more information about this algorithm.

**The match type's distribution logic is not "hard coded".** Because the match type's payment distribution logic is embedded in a plug-in algorithm, you can introduce new algorithms as per your company's requirements.

It's worth noting that payment *distribution* and *freezing* are two separate steps that typically happen in quick succession. The system's standard match event algorithms create the match event during payment distribution. This match event exists in the **open** state (because the payment segment's FT's have not yet been linked to the match event and therefore debit FT's do not equal credit FT's). The **open** match event references the debit FT's (the bill segments and adjustments) for which it pays. It is only at payment freeze time that the credit FT's (the payment segments) are linked to the match event thus allowing the match event to become **balanced**.

If, at freeze time, the payment's credit FT's do not equal the debit FT's on the match event, the match event is left in the **open** state. An alert will appear on Control Central to highlight the existence of **open** match events (if the appropriate alert algorithm is plugged in the installation record). In addition, you can also set up a ToDo entry to highlight the existence of open match events.

## How Are Match Events Cancelled?

A user can cancel an **open** or **balanced** match event at any time. When a match event is **cancelled**, the event's FT's again affect arrears (and they can be associated with new match events). In other words, when a match event is **cancelled**, its FT's are released from the match event and become open-items.

In addition to manual cancellation, the system may automatically cancel a match event when the last of its payment FT's, if any, is cancelled (if you plug-in the appropriate FT freeze plug-in on your open-item account types).

For example, consider a match event that was created when a payment was made. If the payment is subsequently cancelled, the match event is also cancelled (thus releasing the match event's FT's) if no other payment FT's are linked to the match event. Please be aware that FT cancellation also causes a new match event to be created. This match event matches the original FT (the payment segment) and its cancellation FT. This means that the only "open items" that will exist after a payment is cancelled are the debit FT's that were originally paid.

**Reopening bills associated with automatic payment taxpayers.** While many payments are cancelled due to non-sufficient funds, please be aware that if you reopen a bill for which an automatic payment was created, the system will cancel the associated payment. If this payment is associated with a match event (because the account is an open-item account), the match event will be cancelled and a new match event will be created to match the original automatic payment with its cancellation details. This is necessary because a new payment will be created with the bill is subsequently completed and this payment's FT's will be matched to the bill's FT's.

**Canceling a payment can result in many match events being created.** If a cancelled payment has multiple payment segments, a separate match event will be created for each payment segment.

While payment cancellation is the most common type of FT cancellation, be aware that bill segment or adjustment cancellation may also cause a new match event to be created. We don't necessarily want to always link the cancellation FT and its original FT to the same match event. For example, when the cancellation FT is swept on to the next bill it affects the next bill and not the original FT's bill. For cancellations that will not be swept on to the next bill (payment cancellation, cancellation of an adjustment that is not shown on bill, and bill segment cancellation before the bill is completed) the system creates a new match event that matches the original FT and its cancellation FT. This way, neither FT affects aged debt. If the original FT was linked to an existing match event and no other FTs are left on this match event it is automatically canceled.

### Current Amount Is Matched, Not Payoff

The system matches the current amount of financial transactions, not the payoff amount.

Please refer to [Current Amount versus Payoff Amount](#) for more information about current and payoff amounts.

## Disputing Items

Open-item taxpayers may dispute FT's that they are not comfortable paying. For example, a taxpayer who receives a bill with an anomalous charge may decide to dispute it.

When an open-item taxpayer disputes a charge, a user creates a match event and links the disputed FT(s) to it. This match event will be in the **open** state (because it does not contain FT's that sum to zero). In addition, the match event's "disputed switch" is turned on.

**Alerts.** An alert is displayed on control central to highlight the existence of disputed match events (if the appropriate alert algorithm is plugged in). In addition, you can also set up a ToDo entry to highlight the existence of disputed match events.



While the dispute is being researched, the disputed amount will not affect aged debt, but it still forms part of the taxpayer's balance.

If the dispute goes in your company's favor, the disputed match event should be **cancelled** (thus allowing the FT's to again impact aged debt).

If the dispute goes in the taxpayer's favor,

- You may decide to cancel the offending bill segment(s) / adjustment(s). As described above, these cancellations are going to be swept on to the next bill. The system therefore will not automatically cancel the disputed match event. Notice that the cancellation effect of the disputed items is carried over on to the next bill. This means that the previously disputed items still need to be paid.

**Cancel / rebill.** If you cancel / rebill an offending bill segment, both the cancel and the rebill will become open-items that will be matched when the next bill is paid.

- You may decide to issue an adjustment to counter the effect of the disputed FT's. In this situation, you would simply link the adjustment FT to the disputed FT's (thus allowing the match event to become **balanced**). It is important to use in this case an adjustment that does not show on bill.

## Overpayments

An overpayment, by definition, does not "match" to open items. However, the match type algorithms supplied with the base package will result in a **balanced** match event if an overpayment is made. The following points explain how this is achieved:

- The base package's match type algorithms will distribute the payment until the taxpayer's current debt is satisfied.
- The amount of the overpayment will be kept on a separate obligation (this only happens if you plug-in the appropriate Overpayment Distribution algorithm on your account types). Refer to [Overpayment Obligations](#) for more information.
- When the payment is frozen, the payment segments that satisfy current debt will be matched against their respective open-items. The payment segment used to book the overpayment (on the overpayment obligation) will not be matched.
- When future bills are completed, the credit balance on this "overpayment obligation" will be transferred to the "real obligation's" when future bills are completed (if you have plugged in the appropriate bill completion algorithm on the overpayment obligation's obligation type). If the overpayment satisfies all newly calculated charges, a match event is created that matches the new charges against the funds transferred from the overpayment obligation. Refer to [When Are Match Events Created](#) for information about how the system creates match events at bill completion time when the new charges on the bill are satisfied by other credits such as overpayments.
- At some point in the future, the overpayment will be exhausted (i.e., all funds will be transferred to "real obligation's"). At that point in time, the overpayment obligation will close (assuming you set up the overpayment obligation's obligation type as a "one time"). At close time, the system creates a match event that matches the original overpayment payment segment with the adjustments that were used to transfer funds to the "real obligation's". Refer to [When Are Match Events Created](#) for information about how the system creates match events when an obligation closes.



## Setting Up The System To Enable Open Item Accounting

The following section provides an overview of how to enable open-item accounting.

### Contents

- [Match Type Setup](#)
- [Match Event Cancellation Reason Setup](#)
- [Account Type Setup](#)
- [Overpayment Obligation Type Setup](#)
- [Installation Record Setup](#)
- [ToDo Entry Setup](#)

### Match Type Setup

The number of match types that you will need is dependent on the number of ways you want payments to be matched to open items. At a minimum, you will probably need the following match types:

- **Bill ID.** This match type should reference an override payment distribution algorithm that distributes the payment based on the bill ID specified on the payment (in match value). Refer to [Payments And Match Events](#) for more information.
- **Obligation ID.** This match type should reference an override payment distribution algorithm that distributes the payment based on the obligation ID specified on the payment (in match value). Refer to [Payments And Match Events](#) for more information.
- **Pay Plan.** This match type should NOT reference an override payment distribution algorithm (if this algorithm is blank, the account type's payment distribution algorithm is used). Refer to [Pay Plans](#) for more information.

### Match Event Cancellation Reason Setup

The number of match event cancellation reasons that you will need is dependent on the number of ways your organization can justify the cancellation of a match event. At a minimum, you will probably need the following match event cancellation reasons:

- **FT Cancellation.** This cancel reason should be referenced on the account type FT Freeze algorithm that is responsible for canceling match events when one of its financial transactions is cancelled.
- **Incorrect Allocation.** This cancel reason should be specified by users when they cancel match events that were created by the system erroneously.

### Account Type Setup

The following points describe [account type](#) oriented set up functions:

- Turn on the open-item accounting switch.
- Set up the following algorithms for each division:
  - Specify a **payment freeze** algorithm that causes a payment's FT's to be linked to the match event that was created when the payment was distributed. Refer to [Payments And Match Events](#) for more information.
  - Specify a **FT freeze** algorithm that causes match events to be cancelled (and a new match event to be created) when a FT is cancelled. Refer to [How Are Match Events Cancelled](#) for more information about cancellation.

- We strongly recommend specifying an **overpayment** algorithm that causes overpayments to be segregated onto an "excess credit / overpayment" obligation. Refer to [Overpayments](#) for more information.

## Overpayment Obligation Type Setup

Specify a **bill completion** algorithm that causes the credit amount on overpayment obligation's to be transferred to newly create debt (created when the bill is created). This algorithm transfers an overpayment obligation's balance to regular obligation's and creates a match event if the overpayment covers the entire bill. Refer to [Overpayments](#) for more information.

## Installation Record Setup

Specify an **automatic payment** algorithm that causes a match event to be created when automatic payments are created for open-item accounts. The base package algorithm will do this for you if you specify the appropriate parameter on the algorithm. Refer to [APAY-CREATE](#) for more information about this algorithm.

If you want a Control Central alert to highlight when the current account has any open match events, plug in the appropriate **control central alert** algorithm on your installation record. Refer to [C1-OPN-MEVT](#) for more information about this algorithm.

If you want to enable manual pay segment distribution for open item accounts, along with other functions, you will need to plug in an installation algorithm for bill balance calculation. Refer to [C1-OI-BI-AMT](#) for more information about this algorithm.

## ToDo Entry Setup

Two ToDo types are supplied with the base package:

- **TD-MODTL**. This ToDo type highlights the presence of open, disputed match events.
- **TD-MONTL**. This ToDo type highlights the presence of open, non-disputed match events.

Each of the above ToDo types should be configured with the roles that work on entries of each type.

In addition, the account management group and/or divisions from which the default roles are extracted should be updated to define the role that should be defaulted for each of the above ToDo types.

Refer to [The Big Picture Of To Do Lists](#) for more information about ToDo lists.

## Setting Up Match Types

Most payments are distributed amongst obligations using the payment distribution algorithm specified on the payment's account's account type. This algorithm decides how to distribute a payment amongst an account's existing debt if the taxpayer doesn't specify how the payment should be distributed.

A taxpayer can specify how a payment is distributed by specifying a match type and match value on their payments. Consider the following examples:

- Taxpayers that are subject to open-item accounting (this is defined on the account's account type) tell the system exactly which debt is covered by their payments. For example, an open-item taxpayer might make a payment in respect of bill ID **123919101919**.
- Even non open-item taxpayers can direct payments to specific obligations. For example, the system allows a balance-forward taxpayer's payment to be directed to a specific obligation (however, they cannot direct payments to specific bills as only open-item taxpayers can do this).

Match types are used to define the specific type of debt that is covered by a payment. The match type contains the algorithm that effectively overrides the standard payment distribution algorithm defined on the account's account type.

**Background information.** Please refer to [Payments And Match Events](#) and [Match Type Setup](#) for more information about how match types are used.

To set up match types, select **Admin Menu, Match Type**.

#### Description of Page

Enter an easily recognizable **Match Type** and **Description**.

Define the **Pay Dist Override Algorithm** used to distribute payments that reference this match type. If you haven't done so already, you must set up this algorithm in the system. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that overrides the normal payment distribution algorithm.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_MATCH\\_TYPE](#).

## Setting Up Match Event Cancellation Reasons

When a match event is canceled, a cancel reason must be supplied.

**Background information.** Refer to [How Are Match Events Cancelled?](#) and [Setting Up Match Event Cancellation](#) for more information about cancellation.

To set up match event cancellation reasons, select **Admin Menu, Match Event Cancel Reason**.

#### Description of Page

Enter an easily recognizable **Match Event Cancel Reason** and **Description**.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_MEVT\\_CAN\\_RSN](#).

## Fund Accounting

---

The topics in this section provide background information about fund accounting.

**This section is only relevant for some organizations.** The system configuration requirements described in this section are only relevant if your organization practices fund accounting. If your organization does not practice fund accounting, you need only indicate such on the [Installation Record](#); no other setup is required.

### Contents

- [Fund Accounting Overview](#)
- [Accounting Method Is Defined On The Installation Options](#)
- [Fund Controls Fund-Balancing Entries](#)
- [Building Fund-Balancing GL Details](#)
- [Setting Up The System To Enable Fund Accounting](#)

## Fund Accounting Overview

Regulations or other restrictions may require some organizations to account for the finances of each of its departments as a separate entity.

To track the finances of departments separately, the organization sets up a fund for each department. A fund is an accounting entity with its own self-balancing set of accounts. Each fund has its own "sub general ledger" with its own chart of accounts, and within each fund, its debits equal its credits at all times. This allows the organization to report on the financial state of each fund independently.

In addition to having a fund for each department, there is also a general fund, which is used to handle inter-fund transfers as well as shared accounts.

A single obligation may have debt related to different funds. For example, tax liability and penalty and interest may go to the same "revenue" fund, and a collection fee may go to a "collection" fund.

In fund accounting, debits and credits must balance for the whole general ledger and debits and credits within each fund must balance.

### Contents

- [Fund Accounting Example](#)
- [An Example Of A Bill Segment That References Multiple Funds](#)

## Fund Accounting Example

Consider an obligation that owes tax liability and then incurs a collection fee. Note that penalty and interest would also accrue, but are not shown in this example.

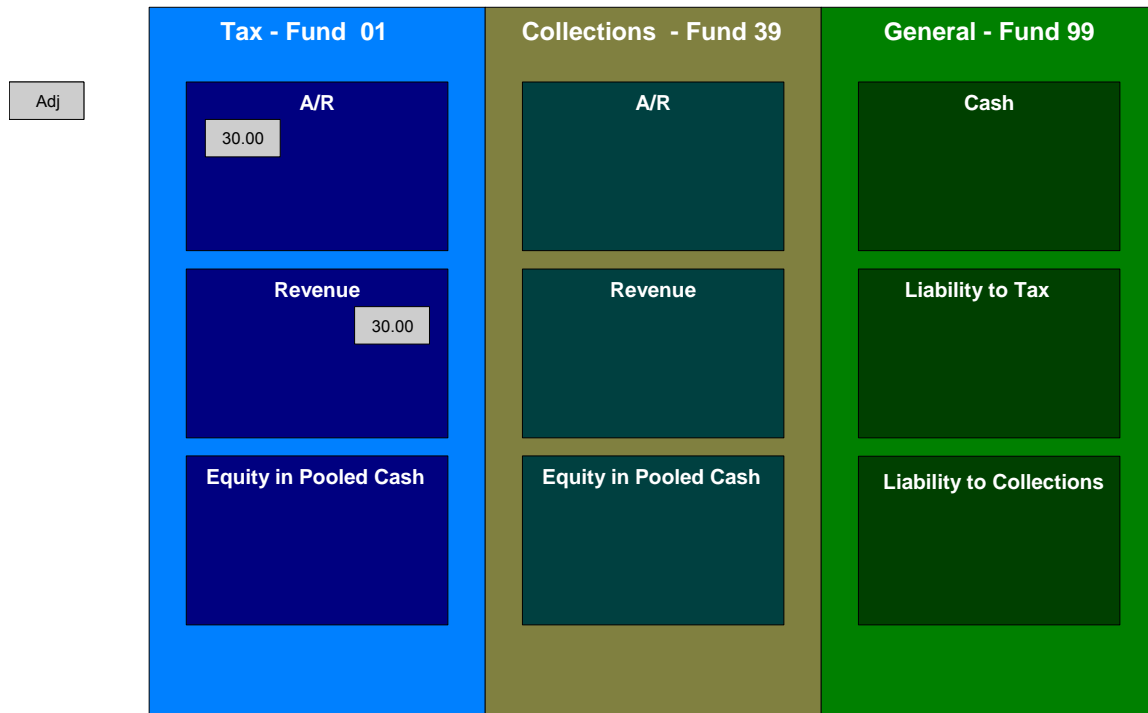
The accounts receivable (A/R) distribution code to use for financial transactions that affect A/R is linked to the obligation type. The assumption is that the fund for this distribution code is the same "revenue" fund.

For this example, three funds exist:

- Tax (fund 01)
- Collections (fund 39)

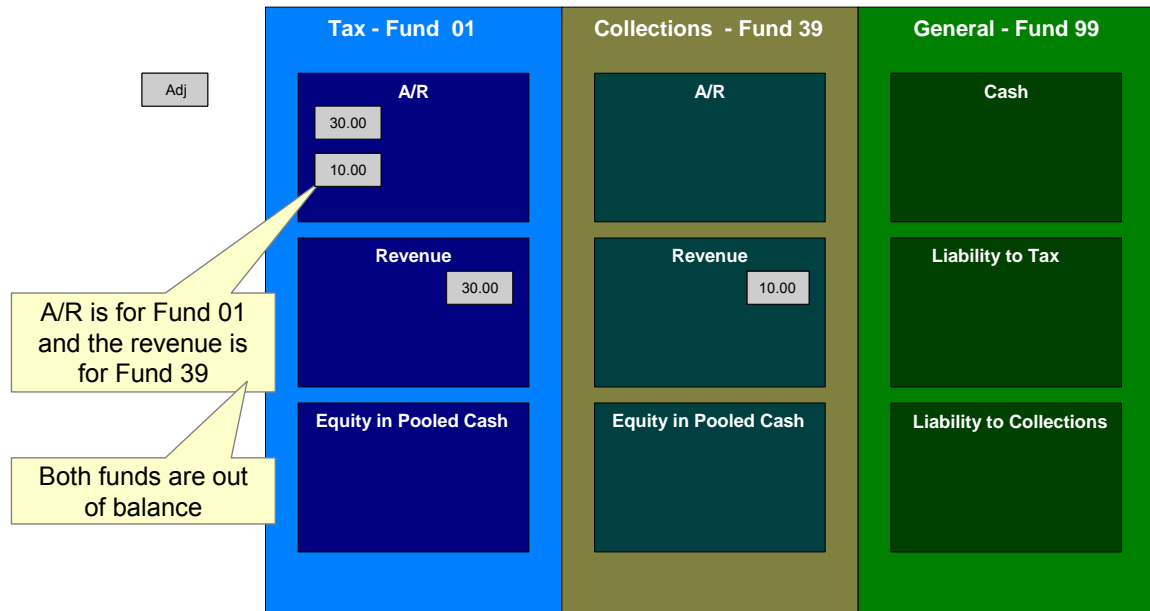
- General (fund 99)

When a tax is levied, the following adjustments are made to the tax fund account producing the following GL entries.



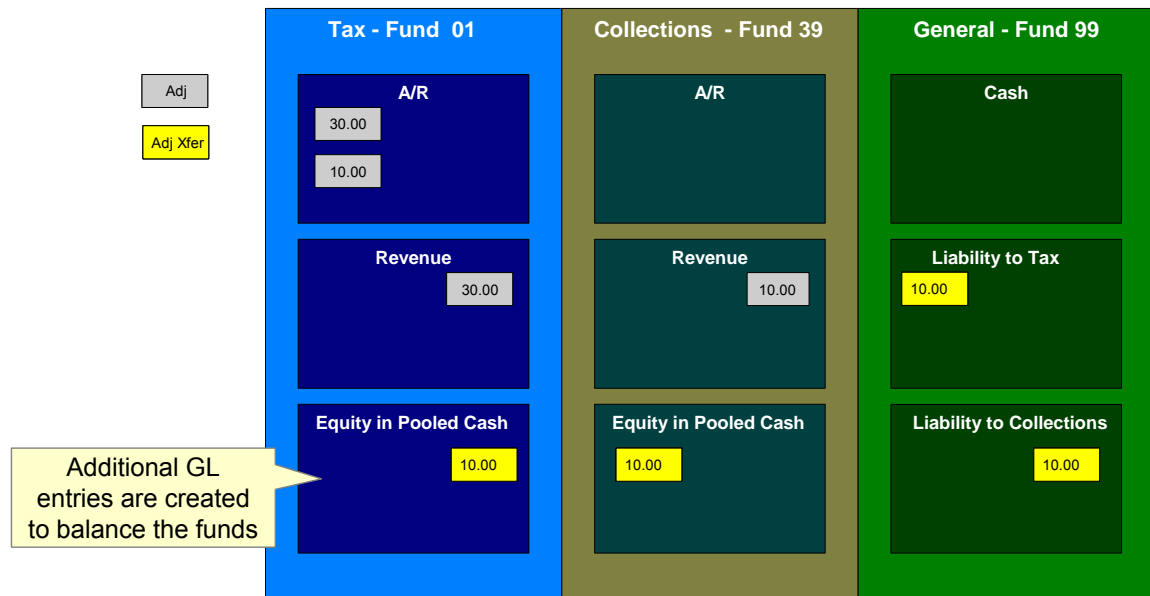
For the tax fund, the GL details of the bill include a debit to the accounts receivable (A/R) account and credits to the revenue account. The taxing authority is owed the entire portion of the adjustment by the taxpayer. The tax fund is balanced.

The following diagram illustrates the initial GL accounting that occurs when a collection fee is levied.

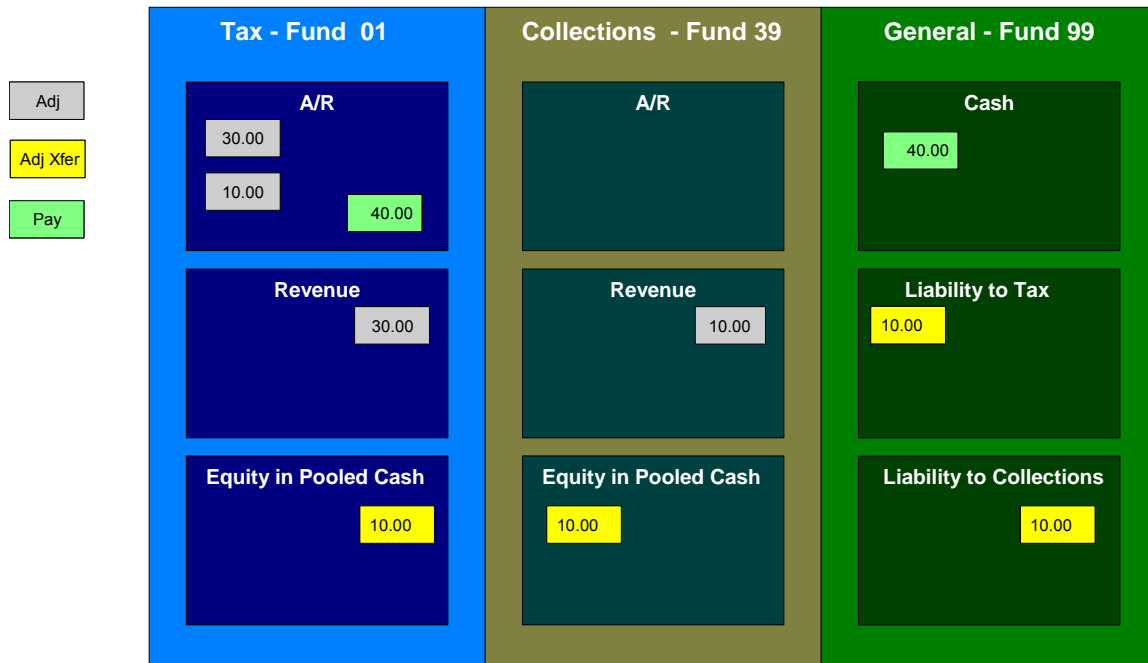


Because A/R is added to the tax fund and the revenue is added to the collections fund, the funds are out of balance.

As the tax authority takes on the responsibility to collect the fee, additional entries are created to balance the funds.



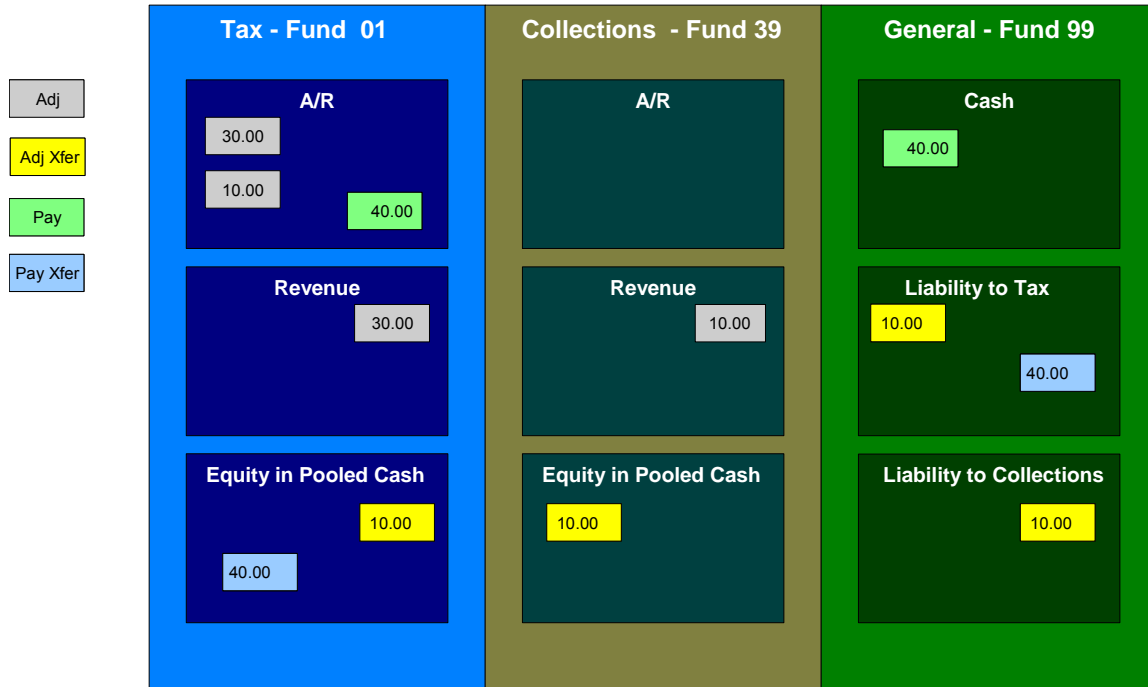
The following diagram illustrates the initial GL accounting that would occur when the payment arrives.



The tax authority's general cash account is debited, and the tax fund's A/R account is credited.

If the accounting were left in this state, the fund accounting principal – that each fund represents an independent entity with a self-balancing chart of accounts – would be violated. This violation is caused due to the fact that cash is recorded on the general fund, not the tax fund, causing the general fund to have an excess debit and the tax fund to have an excess credit.

From an organizational viewpoint, to make the tax fund whole, the tax fund needs to note what portion of the cash it owns, and correspondingly, the tax authority needs to note what portion of the cash is owed to each fund. The following diagram illustrates this point.



To maintain a balance of debits and credits within each fund, the tax and collection funds have an "equity in pooled cash" (EPC) account and the general fund has a liability account for each fund. In addition to debiting the general fund's cash account and crediting the tax fund's A/R accounts, the tax and collection funds' EPC accounts are debited and the general funds liability accounts are credited.

And so, with the additional GL entries, all funds have matching debits and credits.

### An Example Of A Bill Segment That References Multiple Funds

Consider a tax liability that is distributed to multiple jurisdictions where 60% is allocated to the county where the store is located and 40% is allocated to the state.

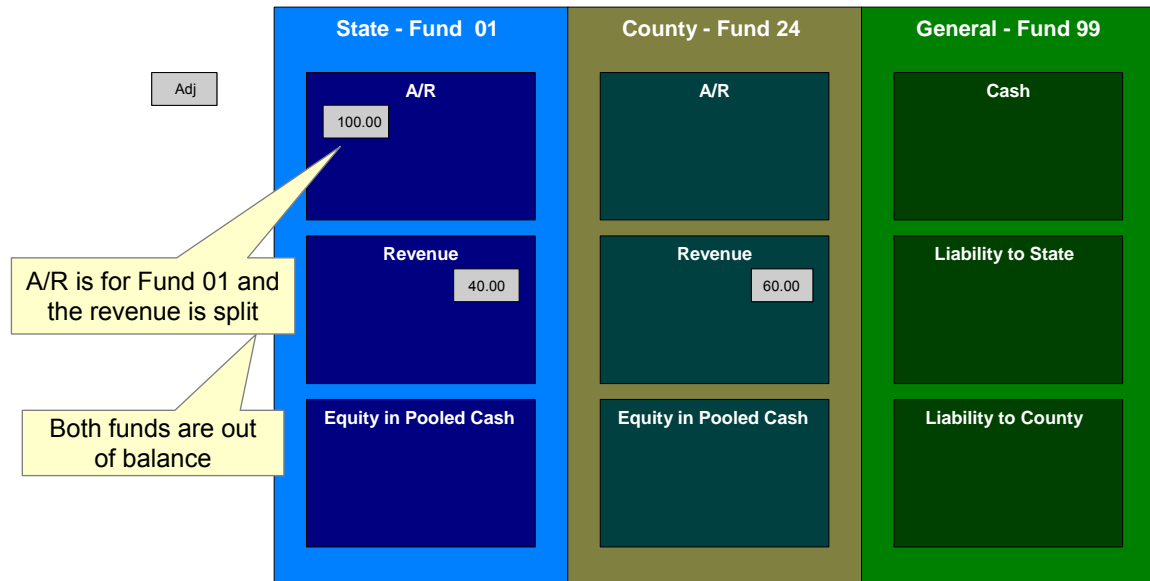
Note that in this example only two jurisdictions are used; however, a real scenario may break this down further into town, school districts, and so on.

For this example, three funds exist:

- State (fund 01)
- County (fund 24)
- General (fund 99)

When a \$100.00 tax is levied, the following diagram illustrates the initial GL entries for this scenario.

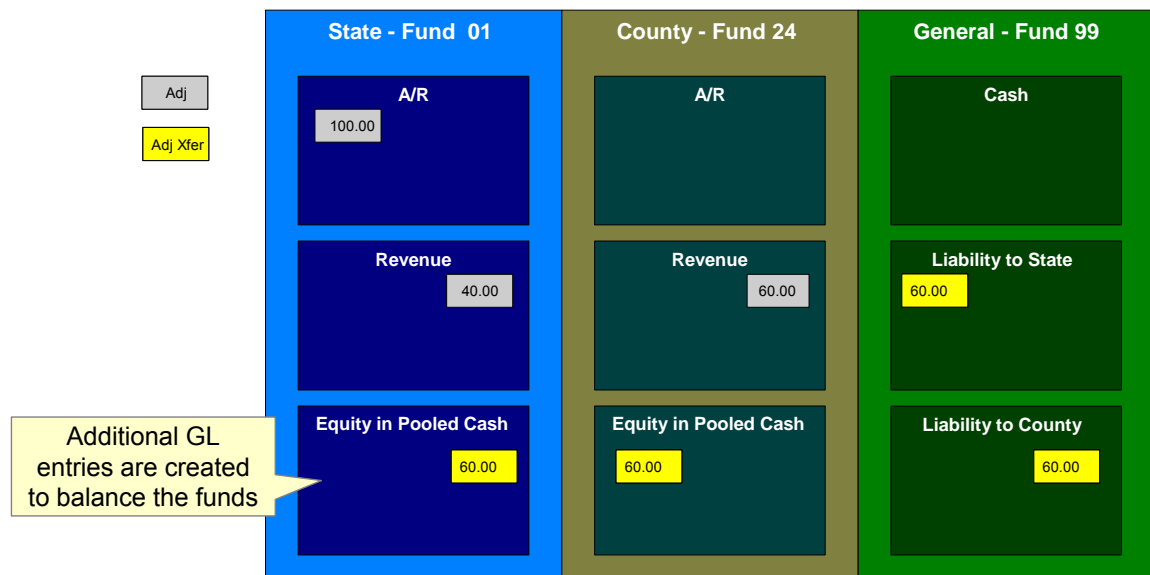




The state fund's A/R is debited, and the state and county funds' revenue accounts are credited.

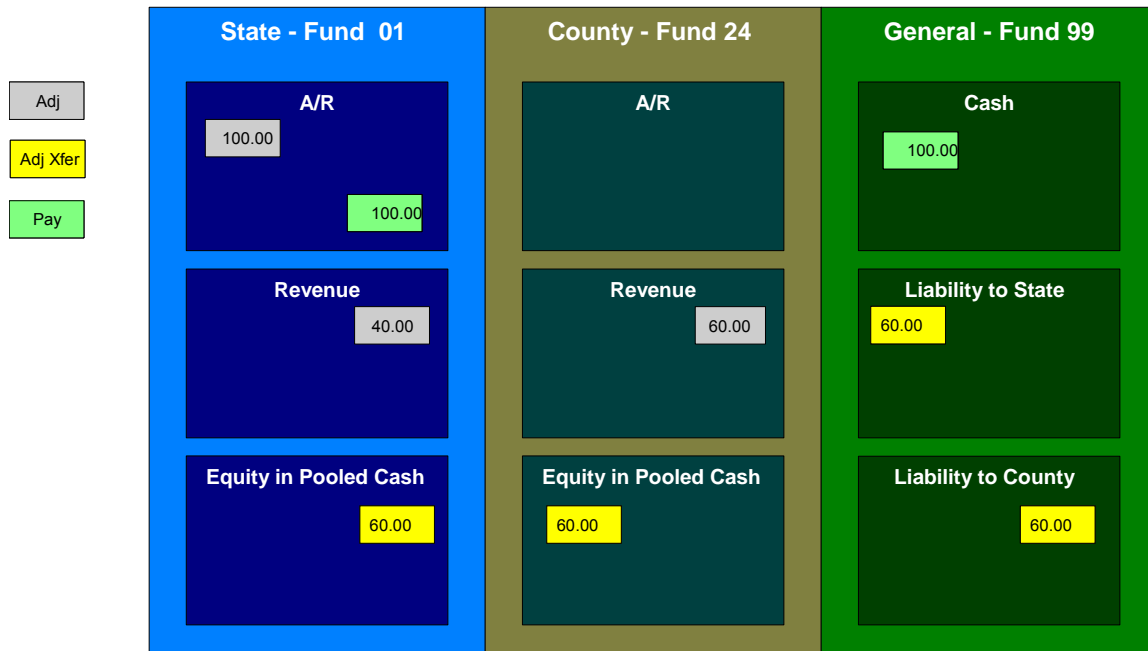
If left at this, the funds would be out of balance; the state fund would have an overall excess debit and the county fund would have an excess credit.

To balance the funds, the state fund accepts the responsibility for collecting the county taxes from the taxpayer but immediately remunerates the charges to the county fund.

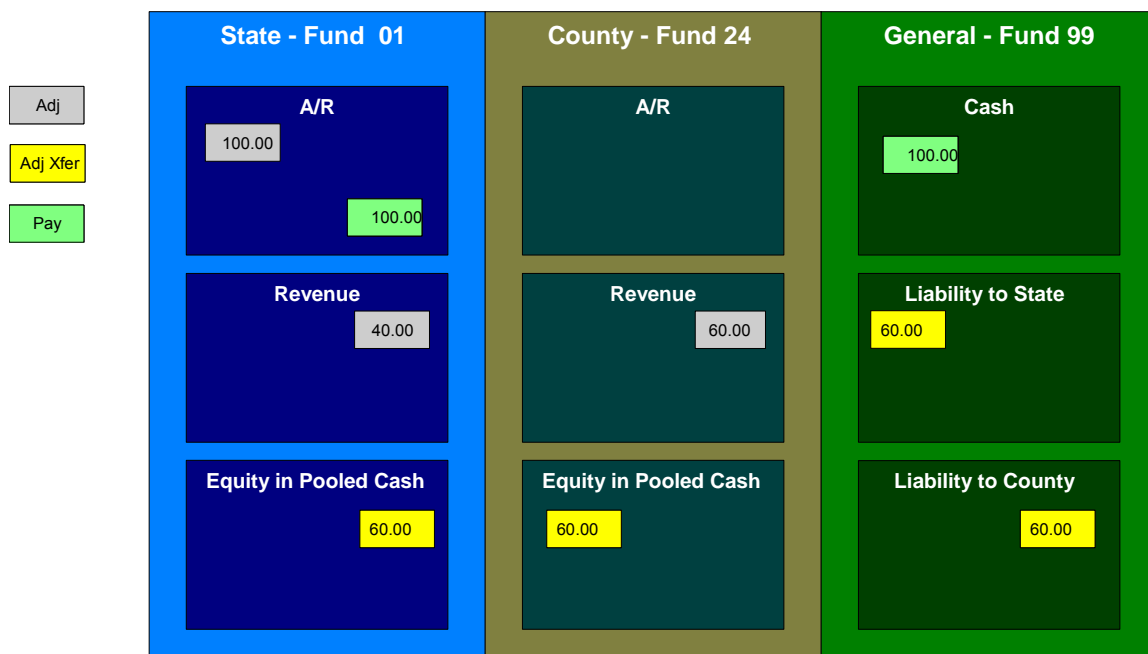


This transfer is done using the general fund. The state fund's EPC account is credited and the liability to state is debited with the amount of the county sales tax revenue. Also, the county fund's EPC account is debited and the general fund's liability to the county account is credited by the county sales tax revenue. In effect, the state fund owes the county charges to the general fund, and the general fund owes the county charges to the county fund.

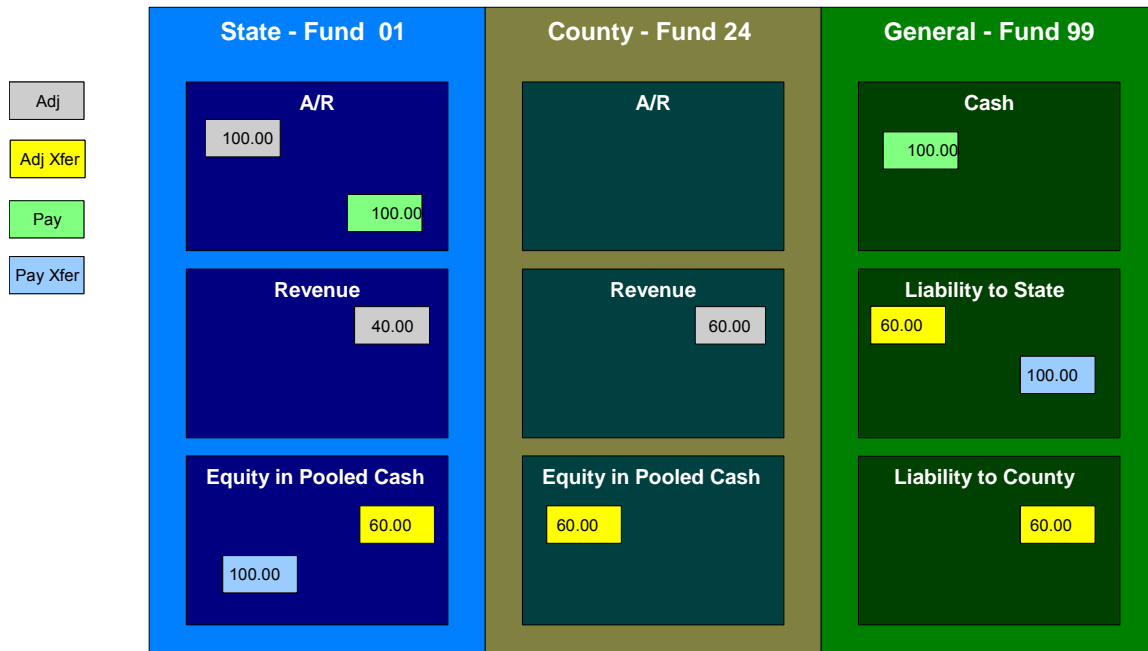
The following diagram illustrates the initial GL accounting that would occur when the payment arrives.



When the payment arrives, the cash is debited to the general fund's cash account, and the state fund's A/R is relieved. Again, the funds would be unbalanced if left in this condition; the state fund would have an excess of credits and the general fund would have an excess of debits.



To maintain each fund's balance of debits and credits, the general fund's liability to the state fund is credited by the amount of the fund's share of the cash, and the state fund's EPC is debited. Note that the payment has no effect on county fund's EPC and the general fund's liability to the county fund. The county fund "received" its money from the state department when the adjustment was created.



And so, all funds have matching debits and credits.

## Accounting Method Is Defined On The Installation Options

You must turn on a switch on the [Installation Record](#) to enable fund accounting.

## Fund Controls Fund-Balancing Entries

There are two levels of debit and credit balancing in fund accounting. There is the balancing required by double entry accounting: the total debits in the entire GL must equal the total credits. This is required regardless of whether fund or corporate accounting is used. The distribution codes for these entries come from varying sources, depending on the type of financial event.

Refer to [The Source Of GL Accounts On Financial Transactions](#) for information on the sources of the distribution codes.

The second level of balancing is specific to fund accounting. Within each fund—not just across the GL—the total debits must equal the total credits. The original distribution code from the financial event has a fund specified. For example, a bill would cause a debit to a fund's A/R distribution code, and included in that A/R distribution code is the fund. It is the definition of the fund that specifies whether fund-balancing entries are required and provides the distribution codes for these entries.

For a departmental fund, the fund-balancing debit and credit would be specified. When a debit is applied to a departmental fund's GL account, an additional account (typically the general fund's liability to the departmental fund) is debited and an account (typically the departmental fund's EPC) is credited. When a credit is applied to a departmental fund's account, an additional account (typically the general fund's liability to the departmental fund) is credited and an account (typically the department's EPC) is debited.

For the general fund, no fund-balancing debits and credits are specified.

## Building Fund-Balancing GL Details

Building the GL details for a financial event is a two-step process.

- First, the system generates the regular GL details for a financial transaction (FT). This is done regardless of whether corporate or fund accounting is used.
- Second, with fund accounting activated, the system analyzes the distribution code on each GL detail associated with the FT. If a [fund](#) is specified on a distribution code, the system checks the definition of the fund. If fund-balancing entries are specified on the fund, two additional GL entries are added to the FT:
  - An offsetting entry to the Equity in Pooled Cash account is created for the departmental fund (e.g., if the FT is debiting a given fund, an offsetting credit is created in the funds EPC account).
  - Another entry to the departments liability account is created for the general fund.

The result is a consolidated set of GL entries for the FT, incorporating the regular entries as well as the fund-balancing entries.

## Setting Up The System To Enable Fund Accounting

The following section provides an overview of how to enable fund accounting.

### Contents

[Turn On Fund Accounting](#)

[Defining Funds](#)

[Distribution Codes Must Include Fund ID](#)

[Update Your Funds With Their Respective Equity and Liability Distribution Codes](#)

### Turn On Fund Accounting

On the [Installation Record](#), indicate that fund accounting is **Practiced**. This is the default setting.

### Defining Funds

A fund must be setup for each specific fund in your organization. Don't forget to also set up a fund for the general fund. Navigate using **Admin Menu, Fund**.

#### Description of Page

Enter a **Fund** and a **Description** to identify the fund.

If this fund is used to balance other funds or to hold cash, indicate a **Fund Type** of **General**, otherwise indicate that it is **Specific**.

If the fund type is **Specific**, specify the **Equity Distribution Code** and **Liability Distribution Code**. These codes are used to balance financial transactions that span funds. The equity distribution code should belong to the same **Fund** as the one you are defining. The liability distribution code should belong to the general fund.

## Distribution Codes Must Include Fund ID

All of your distribution codes must include their respective fund ID.

**For more information**, refer to [Setting Up Distribution Codes](#).

## Update Your Funds With Their Respective Equity and Liability Distribution Codes

After distribution codes have been setup, you must update your funds to indicate the equity and liability accounts used to balance inter-fund financial transactions.

## Defining Bill Cycles and Bill Periods

This section discusses issues related to defining bill cycles and bill periods in the system.

An account may reference a bill cycle. An account's bill cycle may be used to control cyclical billing.

In this section, we describe how to design and set up this cycle. In addition, we discuss how to set up bill period schedules. These are used to define the bill segment end date for special types of obligations.

**Recommendation.** We recommend reading [Bill Frequency – Bill Cycle vs Bill Segment Duration](#) before setting up this information.

### Contents

- [The Big Picture Of Bill Cycles and Bill Periods](#)
- [Setting Up Bill Cycles](#)
- [Setting Up Bill Periods](#)

## The Big Picture Of Bill Cycles and Bill Periods

The topics in this section provide background information about a variety of bill cycle and bill period issues.

### Bill Cycles

The topics in this section provide background information about a variety of bill cycle features.

### Contents

- [The Cyclical Billing Process & Window Billing](#)
- [Designing Your Bill Cycles](#)
- [How Does An Account Get Its Bill Cycle?](#)
- [Protecting An Account's Bill Cycle](#)

#### The Cyclical Billing Process & Window Billing

The cyclical bill creation process creates most bills. This process works as follows:

- An account can reference a bill cycle. The bill cycle's schedule controls WHEN a cyclical billing process attempts to create bills for the account.

- Every bill cycle has a bill cycle schedule that defines the dates when a cycle's accounts are to be billed. Rather than attempt to create bills on one evening, your implementation may use a concept of "window billing" where the system attempts to produce bills for accounts over a few nights. This concept is useful if your billing is based on information being sent from an external source. It allows you to start billing accounts on the earliest possible day and then bill the stragglers over successive evenings. This results in much better cash flow.
- When the bill cycle creation process runs, it looks for bill cycles with open bill windows. It then attempts to create bills for the accounts in each such cycle. If a bill is successfully created, it is completed immediately and ready to send to the taxpayer. If a bill cannot be created, the system will create a bill in the "error" state and initiate a To Do entry with a message that can be analyzed by your billing staff. When the bill cycle creation process next runs, it deletes all "error" bills and attempts to recreate them. It continues this throughout the bill window. If bills are in error at the end of the window, they will remain in this state until a user fixes them. If the bills are still in error when the cycle's next window opens, a billing error will be generated.

### Designing Your Bill Cycles

The number of bill cycles is determined by how much you want to stagger the billing of your taxpayers. You may do this to smooth out calls you may receive for your call center with questions about tax bills.

### How Does An Account Get Its Bill Cycle?

Most accounts are created behind-the-scenes through taxpayer registration processing. An account created like this doesn't have a bill cycle. Rather, it sits in limbo awaiting the activation of its first obligation that should be billed using cyclical billing. When an obligation is activated, an activation algorithm should assign the account to an appropriate cycle based on business rules.

Note, an account may be configured to protect its bill cycle from being changed by an algorithm. Refer to [Protecting An Account's Bill Cycle](#) for more information.

**A To Do entry highlights accounts without a bill cycle.** A To Do entry highlights those accounts that require a user to specify a bill cycle. This entry is generated for accounts without a bill cycle that have active obligations.

### Protecting An Account's Bill Cycle

An account has a single bill cycle, but may have multiple obligations that are billed via cyclical billing process. In this case, when a cyclical obligation is activated, you may not want an algorithm to change the account's bill cycle. To prevent this you need to turn on the account's **protect bill cycle** flag.

When the last obligation for an account is stopped, the protect bill cycle flag is reset. This is to ensure that if the taxpayer starts a new obligation in the future the bill cycle can be set based on the new obligation's activation algorithm rules.

## Setting Up Bill Cycles

An account may reference a bill cycle. The bill cycle defines the schedule for cyclical billing of its accounts. To define a bill cycle and its bill cycle schedule, open **Admin Menu, Bill Cycle**.

**Description of Page**

Enter a unique **Bill Cycle** and **Description** for every bill cycle.

Use the **Bill Cycle Schedule** collection to define when bills are produced for the accounts in a given bill cycle. The following fields are required for each instance:

<b>Window Start Date</b>	Specify the date on which the system should start trying to create bills for accounts in the cycle.
<b>Window End Date</b>	Specify the last day on which the system will create bills for accounts in the cycle.
<b>Accounting Date</b>	Specify the financial date associated with the bills' financial transaction. The accounting date defines the financial period(s) to which the bills will be booked in your general ledger.
<b>Freeze and Complete</b>	Turn on this switch if the system should freeze and complete any bill that is created without errors. If this switch is turned off, all bills created by a cyclical billing process should be left in the unfinished state. You would only turn this switch off if you want to verify an entire bill run prior to freezing it (e.g., if you are introducing a new version of a rate). If you turn this off, you will need to return to this page after verifying a bill run and turn it back on for the taxpayers to receive bills. When the cyclical billing process next runs, it should delete all unfrozen bills and recreate them as per the instructions on the bill cycle schedule.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI BILL CYC](#).

## Setting Up Bill Periods

Some obligation types reference a bill period. The bill period defines when the obligation's bill segments are produced and the respective end date of each bill segment.

To define a bill period and the bill period schedule, open **Admin Menu, Bill Period**.

**Description of Page**

Enter a unique **Bill Period** and a **Description** for every bill period.

Use the **Bill Period Schedules** collection when the system should create bill segments for obligations that use a given bill period. It also defines the end date of each respective bill segment. The following fields are required:

<b>Bill Date</b>	Specify the earliest date on which the system is allowed to create a bill segment for obligations using this bill period.
------------------	---

**Bill Seg End Date**

Specify the end date of the bill segment. For future bills, this will be after the bill date. For retro bills, this will be before the bill date.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI BILL PERIOD](#).

This information is used by the bill segment creation process to determine the end date of obligations that use a bill period.

## Other Financial Transaction Topics

---

Various topics about financial transactions are discussed in this section.

### The Source Of GL Accounts On Financial Transactions

The following table lists the major financial events, their standard accounting, and the source of distribution codes used to derive the GL accounts sent to your general ledger.

Financial event	GL Accounting	Source Of Distribution Code
Create a normal bill segment. <i>Bill Segment FT Algorithm is Payoff Amt = Bill Amt / Current Amt = Amt Due</i>	Debit: A/R	Obligation Type
	Credit: Revenue	Rate Component
Create a bill for company usage. <i>Bill Segment FT Algorithm is Payoff Amt = 0 / Current Amt = 0</i>	Debit: Company Usage Expense	Obligation Type
	Credit: Revenue	Rate Component
Create a bill for charity. <i>Bill Segment FT Algorithm is Payoff Amt=0 / Current Amt = Bill Amt</i>	N/A – charity bills have no effect in the GL	N/A
	N/A	N/A
Create a payment segment for a normal obligation	Debit: Cash	Bank Account on the Tender Source of the Tender Control for the Payment Segment's Tender.
	Credit: A/R	Obligation Type
Create a payment segment for a charitable contribution obligation	Debit: Cash	Bank Account on the Tender Source of the Tender Control for the Payment Segment's Tender.
	Credit: Charity Payable	Obligation Type
Create a payment segment for auto-pay at bill completion time	Debit: Cash	Bank Account on the Tender Source on the Auto-pay Route Type of the Auto-pay Source.



Financial event	GL Accounting	Source Of Distribution Code
	Credit: A/R	Obligation Type
Canceling a payment	Debit: A/R	Obligation Type
	Credit: Cash	Bank Account specified by the user on the cancel tender page. Note that this defaults to the original tender's bank account.
Create an adjustment to levy a charge	Debit: A/R	Obligation Type
	Credit: Revenue	Adjustment Type

The bottom line is as follows:

- If a bill segment has a financial effect, the distribution code to debit comes from the distribution code on the obligation type, the distribution code to credit comes from the rate component(s) used to calculate the bill segment.
- Payment segments always have a financial effect; the distribution code to debit comes from the bank account on the tender source of the tender control of the tender, the distribution code to credit comes from the obligation type.
- If an adjustment has a financial effect, the distribution code to debit and credit comes from the obligation type and adjustment type. If the adjustment is positive (i.e., the taxpayer owes your organization more money), the distribution code to debit comes from the obligation type; the distribution code to credit comes from the adjustment type. Vice versa if the adjustment is negative.

# Defining Taxpayer Options

The definition of a taxpayer is someone (or something) with financial obligations with your tax authority. Within the documentation, the terms “taxpayer” and “customer” may be used interchangeably.

The system subdivides taxpayer information into the following records:

- **Person.** The person record holds demographic information about your taxpayers and every other individual or business with which your tax authority has contact. For example, in addition to normal registered taxpayers, person records may include contacts, accountants, related parties, mortgage brokers, related businesses in a corporate structure, overseas entities, persons of interest, and so on.
- **Account.** Accounts are the entities for which bills are produced and therefore you must create at least one account for every person who has financial obligations with your company. The account record contains information that controls when the bills are created and how the bills are formatted.
- **Tax Role.** A tax role is used to define an instance of a specific tax type for a specific account. It includes information that is specific to the tax type for the account, for example, the dates that the tax is applicable for the account and the filing calendar that defines the filing frequency.
- **Obligation.** An obligation is a contract between the tax authority and the taxpayer. For example, an obligation may represent a specific filing period where a taxpayer is expected to file a return for a given tax type. An obligation may represent an ongoing charge such as a tax preparer fee. An obligation may be used to hold miscellaneous financial information such as an excess credit to later be applied to unpaid tax or debt from a third party source.

Before you can define persons, accounts, tax roles and obligations, you must set up the control tables defined in this section.

**Note.** In this section, we limit the discussion to tables that control basic demographic and financial information about your taxpayers. Other sections describe the tables that control other taxpayer related functions like forms processing, penalty and interest and collections.

## Contents

- [Taxpayer Overview](#)
- [Setting Up Person Options](#)
- [Setting Up Account Options](#)
- [Setting Up Customer Contact Options](#)
- [Setting Up Filing Calendars](#)
- [Setting Up Tax Types](#)
- [Setting Up Obligation Options](#)

## Taxpayer Overview

---

This section describes how the person, account, and obligation records are used to record your taxpayers' demographic and billing options.

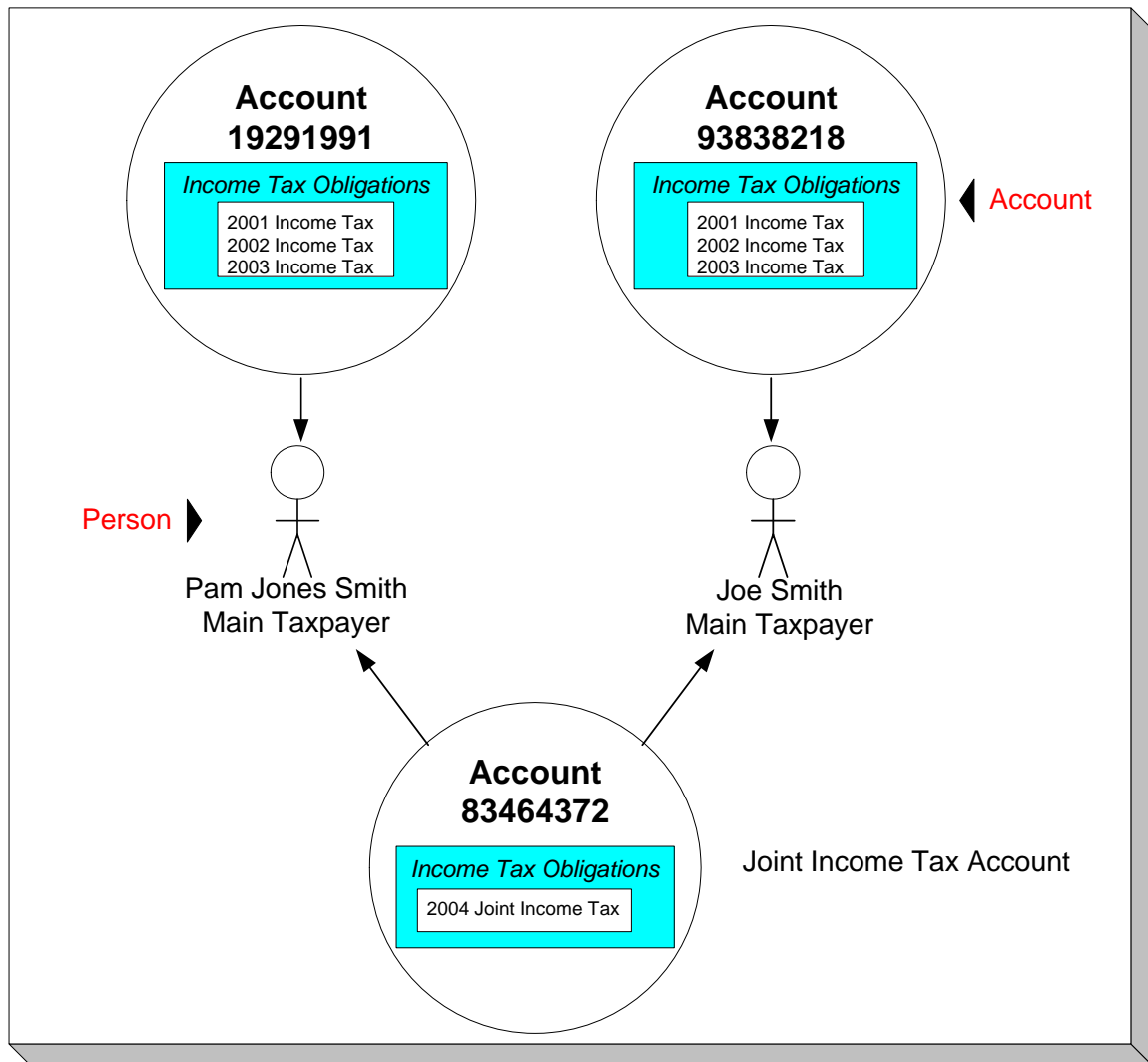
## Contents

## A Simple Example Of Joint Taxpayers

Persons  
Accounts  
Tax Roles  
Obligations

## A Simple Example Of Joint Taxpayers

The following picture illustrates two joint taxpayers: Joe Smith and Pam Jones Smith. Joe and Pam are the "main taxpayers" on their own Income Tax Accounts with their own obligations for tax years 2001, 2002, and 2003. Joe and Pam get married and file a joint tax return for tax year 2004. A new account is established for their 2004 obligation. They are both financially responsible for tax year 2004 while they each remain responsible for their prior tax years. Joe and Pam are each linked to two accounts for tax years 2001-2003 and to the new joint account.



## Persons

Person records hold demographic information about the individuals and businesses with whom your organization communicates. Demographic information includes phone number(s), names and aliases, identification numbers, employment information, etc.

In the above example, 2 person records would be needed, one for Pam Jones and another for Joe Smith.

A new person is added when you first have contact with a person; the person does not have to be a taxpayer before he or she is added. For example:

- If John Doe is the main contact for XYZ Corporation, John might not live in the taxing jurisdiction. You can add John as a new person and link him to any of the XYZ Corporation accounts as a contact (but not financially responsible).
- In situations with non-filers and investigations, you can add a new person and associate it with the case. If the tax authority thinks that ZZZ Corporation is operating as a business but not registered and paying taxes, the tax authority may create a person record at the beginning of the investigation to send correspondence.

**Businesses are persons too.** In addition to humans, you use person records to maintain basic information about the businesses with which your organization has contact.

For a description of the control tables that must be set up before you can define a person, refer to [Setting Up Person Options](#).

## Accounts

The system allows for a taxpayer to define one account for multiple types of taxes. For example, a corporation may have a single account for the corporate income tax, sales and use tax, withholding tax, various excise taxes, etc.

Alternatively separate accounts may be used for each type of tax.

### Contents

- [Account ID Is Non-Intelligent](#)
- [Account / Person Cross Reference](#)
- [When Is An Account Created?](#)
- [When Is An Account Expired?](#)

### Account ID Is Non-Intelligent

The unique number of an account is referred to as the “account ID”. You are probably very comfortable with this concept. You may, however, have difficulty dealing with the fact that the account id in this system has no intelligence built into it (e.g., many systems include the bill cycle and geographic location in the account id). In this system, the account ID is a random, system-assigned value.

Because the account ID contains no meaning, it can remain with a taxpayer for life, regardless of where they live, when they are billed, the type of service they receive, etc. This is important because it means that all of the financial history linked to the account remains with the taxpayer for life.

**Technical note.** The non-intelligence of the account ID is also important from the perspective of the parallel processing that takes place when the system creates bills. Because the collection of accounts to be billed in any given bill cycle will be randomly distributed through the number spectrum, the system can distribute account number ranges to parallel threads and each thread will process roughly the same number of accounts.

## Account / Person Cross Reference

A person may be linked to zero or more accounts. A person won't be linked to an account when they have no financial relationship with your organization. A person will be linked to multiple accounts when they have financial relationships with more than one account.

An account must reference at least one person (i.e., the main taxpayer), but may reference an unlimited number of individuals. Multiple persons are linked to an account when several parties have some type of financial relationship with the account (e.g., third party guarantors, account contact, bill copy recipients, etc.).

## When Is An Account Created?

There are several instances when accounts must be created in the system. An account is created when:

- The taxpayer is registered for a tax obligation, or has any financial interactions with the tax authority.
- The taxpayer is registered for a new tax type. The account is created to track the obligations of the taxpayer to file or pay taxes.
- An estimated payment is received from a taxpayer for a tax that is not yet registered. The account must be created to hold the estimated payment.
- A non-filer is identified, and the tax authority decides to send a default tax assessment or penalty.

## When Is An Account Expired?

Accounts never expire. Once a taxpayer has an account, the account remains in the system forever. Linked to the account are obligation records that define a taxpayer's obligation to file a tax return, and/or make a payment to the tax authority. When an account has obligations, the system pursues unpaid and overdue debt, refunds credits overdue to the taxpayer, and pursues obligations to file returns that have not been fulfilled. If the account doesn't have active obligations, the system will not produce a bill for it. You can think of an account without active obligations as being "dormant", waiting for the day when the taxpayer again files a tax return. If the taxpayer never re-files, the account (along with its financial history) remains dormant forever.

## Tax Roles

Tax Roles represent the ongoing obligations of a given tax type within an account at a given point in time. It may be also be thought of as the reason a taxpayer must interact with the tax authority, or one of the taxpayer's relationships with the tax authority.

Tax roles include the dates that the tax is effective. If you consider a business, some tax types are required for the entire time a company is in business, such as corporate income tax. Other tax types may only be in effect if the company is engaging in certain activities, which may change over time.

### Contents

- [When Is A Tax Role Created?](#)
- [Filing Calendars for Filing Tax Types](#)

### When Is A Tax Role Created?

Most tax roles are created when the tax authority is aware that a tax type is applicable for an account. For example, when a business taxpayer registers their business and indicates the various tax types that are applicable or when an individual taxpayer files a tax form.

### Filing Calendars for Filing Tax Types

For tax types that are filing period based, the onus is on the taxpayer to file the return and pay the appropriate assessment on time. The tax type typically defines the valid filing calendar(s) that govern the frequency of filing and the dates to file for each filing period. For some tax types a single calendar is valid for all tax roles of that tax type. For other tax types, one or more different filing frequencies (and therefore filing calendars) may be applicable and may change over the life of the tax role for an account.

## Obligations

An obligation is a contract (either formal or implied) between the tax authority and a taxpayer. An obligation is linked to an account. There is no limit to the number of obligations that may be linked to an account.

### Contents

- [When Is An Obligation Created?](#)
- [Due Dates for Filing Period Based Obligations](#)
- [Financial Transactions Are Linked To Obligations](#)

### When Is An Obligation Created?

For some tax types there is an ongoing obligation to file, such as sales tax. For obligations related to these types of taxes, it may be beneficial to include logic to create future obligations based on the start date, end date, and frequency of the obligation.

For other tax types, such as individual income taxes and excise taxes, obligation records may be created when the tax authority receives a tax return.

## Due Dates for Filing Period Based Obligations

For tax types that are filing period based, the onus is on the taxpayer to file the return and pay the appropriate assessment on time. This section describes the configuration provided for defining and determining due dates.

When should the taxpayer file the return for a given filing period? When defining filing periods for a [filing calendar](#) you also define the statutory Due Date of the return along with the Grace Date. For taxpayers that request an extension to their filing date, an Override Filing Due Date may be defined on the specific [obligation](#).

Typically the statutory payment due date for a filing period is the same date as the statutory filing due date. However, the dates may be extended independently. Extensions to a payment date are also defined on a specific obligation using the Override Payment Due Date. Also note that the grace date for the payment due date is calculated differently. Instead of using the explicit Grace Date on the filing calendar, the obligation type defines a Payment Grace Days. The logic in the base algorithms that determine if a payment is made on time (whether it be for the statutory due date or the override payment due date) considers the grace days in its calculations.

## Financial Transactions Are Linked To Obligations

For more information about how financial transactions are linked to obligations, refer to [The Financial Big Picture](#).

## Setting Up Person Options

This section describes tables that must be set up before you can define persons.

### Contents

- [Defining Identifier Types](#)
- [Defining Person Relationship Types](#)
- [Defining Person Types](#)

## Defining Identifier Types

When you set up a person, you may define the various types of identification associated with the person, e.g., their driver's license number, their tax identity, etc. Every piece of identification associated with a person has an identification type. These identifier type codes are defined using **Admin Menu, Identifier Type**.

**How are person identifiers used?** The reason why identifiers are defined on a person is so that users you can look for a taxpayer using one of their person identifiers (see [Control Central - Search Facilities](#) for more information). In addition, person identifiers help prevent duplicate persons from being added to the database. This is because the system warns a user before they add a new person when a person exists with the same identifier.

**Person identifier types are optional.** An [installation option](#) controls whether at least one identifier type is required on every person.

**Description of Page**

Enter an easily recognizable **ID Type** and **Description** for the Identifier Type.

If the identifier type has a format against which validation can be performed, use **Identifier Format** to define the algorithm. To do this:

- Create a new algorithm (refer to [Setting Up Algorithms](#)).
- On this algorithm, reference an Algorithm Type that validates identifier types. Click [here](#) to see the algorithm types available for this plug-in spot.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_ID\\_TYPE](#).

## Defining Person Relationship Types

It is possible to associate persons to other person. For example,

- You might want to define the subsidiaries of a parent corporation
- You might want to define spouses as separate persons and then link each person to another person

When you link a person to another person, you must define in what way the person is related to the other person by using a person relationship type code. These codes are defined using **Admin Menu, Person Relationship Type**.

**Description of Page**

Enter the following for each relationship type:

- Enter an easily recognizable **Relationship Type** code.
- Use **Description (Person1=>Person2)** to describe how the first person is related to the second person.
- Use **Description (Person2=>Person1)** to describe how the second person is related to the first person.

**Person1 versus Person 2.** When you link persons together, you do it in respect of one of the persons (which we call Person 1). For example, if you want to link the subsidiaries to a parent company, you do this in respect of the parent company (i.e., you define the parent company's subsidiaries using the [Person - Persons](#) transaction).

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_PER\\_REL\\_TYPE](#).



## Defining Person Types

A person exists for every individual or business with which your tax authority has contact. Besides individual and business taxpayers, persons exist for contractors, accountants at corporate taxpayers, third party guarantors, collection agencies, etc.

The person type is used to define business rules for different types of persons. For example, for a business that is a limited liability partnership, you may wish to require that multiple individual partners be linked to the partnership.

A person type is governed by a [business object](#). The base product provides business objects that you may use or extend. Refer to the BO definitions in the application. Or you may create your own.

Open **Admin Menu, Person Type** to define the person types used to categorize your persons.

The topics in this section describe the base-package zones that appear on the person type portal.

### Contents

- [Person Type List](#)
- [Person Type Actions](#)
- [Person Type](#)

### Person Type List

The Person Type [List zone](#) lists every person type. The following functions are available:

- Click the [broadcast](#) icon to open other zones that contain more information about the adjacent person type.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each person type.

Click the **Add** link in the zone's title bar to add a new person type.

### Person Type Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions are available.

### Person Type

The Person Type zone contains display-only information about a person type. This zone appears when a person type has been broadcast from the Person Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_PER\\_TYPE](#).

## Setting Up Account Options

---

This section describes tables that must be set up before an account can receive a bill.

### Contents

[Setting Up Account Management Groups](#)  
[Setting Up Account Relationship Codes](#)  
[Setting Up Alert Types](#)  
[Setting Up Bill Messages](#)  
[Setting Up Bill Route Types](#)  
[Setting Up Bill Cycles](#)  
[Setting Up Account Types](#)  
[Setting Up Collection Classes](#)

## Setting Up Account Management Groups

Users are informed that something requires their attention by entries that appear in To Do lists. For example, consider what happens when:

- A tax return fails validation for a math error.
- A tax return meets the business rule criteria to trigger a desk audit.
- A taxpayer refund exceeds the business rule review thresholds.
- A letter to a taxpayer is in error because the taxpayer does not have an address record.

You can optionally use account management groups (AMG) to define the respective role to be assigned to To Do entries that are associated with an account and a given To Do type. For example, you can create an AMG called **Credit Risks** and assign this to accounts with suspect credit. Then, whenever an account-oriented To Do entry is created for such an account, it will be assigned a role based on the **Credit Risks** AMG. Refer to [Assigning A To Do Role](#) for more information.

**Account management groups are optional.** You need only set up account management groups (and link them to accounts) if you wish to address specific To Do entries associated with specific accounts to specific roles.

Account management groups are defined using **Admin Menu, Account Management Group**.

### Description of Page

Enter an easily recognizable **Account Management Group** code and **Description** for each account management group. Use the grid to define the **ToDo Role** to be assigned to entries of a given **To Do Type** that are associated with accounts that reference the **Account Management Group**.

**Note.** Only To Do entries that are account-oriented take advantage of the roles defined for an account management group (because only accounts reference an account management group).

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI ACCT MGMT GR](#).

## Setting Up Account Relationship Codes

When you link a person to an account, you must define in what way the person is related to the account by using an account relationship code. These codes are defined using **Admin Menu, Account Relationship Type**.

### Description of Page

Enter an easily recognizable **Relationship Type** and **Description** for each relationship type.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_ACCT\\_REL\\_TYP](#).

## Setting Up Alert Types

Account based alerts that appear in control central have an **Alert Type**. To define valid alert types, navigate to **Admin Menu, Alert Type**.

### Description of Page

Enter an easily recognizable **Alert Type Code** and **Description** for each alert type. Specify the **Alert Days** to indicate the amount of time that alerts of this type will be effective by default. Specify a value of zero to indicate that alerts of this type will be effective indefinitely by default.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_ALERT\\_TYPE](#).

## Setting Up Bill Messages

There are various informational and warning messages that may appear on an account's bills. Each message is identified with a bill message code. To define a bill message code, open **Admin Menu, Bill Message**.

### Description of Page

Enter a unique **Message Code** and **Description** for every bill message.

The following attributes control how and where the bill message appears on the taxpayer's bill:

**Priority** controls the order in which the message appears when multiple messages appear on a bill.

**Note.** The values for this field are customizable using the Lookup table. This field name is MSG\_PRIORITY\_FLG.

**Insert Code** controls whether a document should be inserted into the bill envelope when the bill message appears on a bill.

**Message on Bill** is the actual verbiage that appears on the taxpayer's bill. If the message text is not static (e.g., field values need to be substituted into the body of the message), you can use the %n notation within the **Message on Bill** to cause field values to be substituted into a message. Refer to [Substituting Field Values Into A Bill Message](#) for more information.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI BILL MSG](#).

## Setting Up Bill Route Types

Bill route types define the method used to route bills to accounts. To define a bill route type, open **Admin Menu, Bill Route Type**.

#### Description of Page

Enter a unique **Bill Route Type** and **Description** for every bill route type.

**Bill Routing Method** controls the type of information that may be defined when the respective **Bill Route Type** is selected on [Account - Person Information](#). The following options are available:

- **Postal**. Use this method if the routing is via the postal service.
- **Fax**. Use this method if the routing is via fax.
- **Email**. Use this method if the routing is via email.

**Note.** The values for **Bill Routing Method** are customizable using the [Lookup](#) table. This field name is BILL\_RTG\_METH\_FLG.

The next two fields control how bills that are routed using this method are [printed](#) (both in batch and online).

- Use **Batch Control** to define the background process that performs the actual download of the billing information. Refer to [Technical Implementation of Printing Bills In Batch](#) for more information about these processes.
- Use **Extract Algorithm** to define the algorithm that constructs the records that contain the information that appears on a printed bill. Refer to [Printing Bills](#) for more information.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI BILL RT TYPE](#).

## Setting Up Bill Cycles

Refer to [Defining Bill Cycles](#) for a description of how to set up bill cycles.

## Setting Up Account Types

When you set up an account, you must assign it an account type. The topics in this section describe the account type control table.

### Contents

- [Account Type - Main](#)
- [Account Type - Bill Messages](#)
- [Account Type - Controls](#)

### Account Type - Main

To set up account types, navigate to **Admin Menu, Account Type - Main**.

#### Description of Page

Enter a unique **Account Type** code and **Description** for every account type.

Use **Collection Class** to define the collection class that defaults onto new accounts that belong to this account type. An account's collection class may be subsequently modified if the account has special collection problems or needs.

Use an **Account Business Object** to define a [BO](#) that may govern additional rules related to accounts of this type.

Turn on **Business Activity Required** if obligations linked to accounts with this account type require a Business Activity description to be entered.

Turn on **Open Item Accounting** if accounts belonging to this account type are subject to open-item account. Refer to [Open Item Accounting](#) for a complete explanation of the significance of this switch.

Turn on **Non Tax Agency Payment** if accounts belonging to this account type are used for payments made to reduce non-tax authority debt. For example, if your tax authority sends collection notices or offsets taxpayer refunds on behalf of other agencies, such as child support, college tuition, or parking violations, you should set up the following information to accept such payments:

Create a new account type called "Non Tax Authority Customer".

- Create an obligation type for each type of non-tax authority payment that taxpayers can make. Make sure to enter a distribution code on each obligation type that references the appropriate revenue (or payable) account. Don't forget to indicate that each obligation type is not billed.
- Create an account to which you'll book such payments. Have this account reference the new account type. We recommend creating a separate account for each obligation type that you created in the previous step.
- Create and activate an obligation for the new account(s).

When someone pays for non-tax authority debt, the operator will add a payment for the above account. On the payment, the operator should record reference information in order to know exactly why the payment was made. Refer to [Payment Event - Main](#) for more information.

You must define a variety of business rules for every division in which an account type has taxpayers. You do this on the [Account Type - Controls](#) page. The grid that follows simply shows the divisions for which business rules have been set up.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_CUST\\_CL](#).

**Account Type - Bill Messages**

When an account type has bill messages, the system will sweep these messages onto bills created for accounts belonging to the account type. Use this page to define an account type's bill messages. Navigate to **Admin Menu, Account Type, Bill Messages** tab to maintain this information.

**Description of Page**

Use the bill messages collection to define **Bill Message** codes that should appear on bills that created for accounts that belong to a given account type. For each message, also specify the **Start Date** and **End Date** when such a message should appear on the bill (leave **End Date** blank if the message should appear indefinitely).

**Where Used**

The system snaps account type bill messages on a bill during bill completion. For more information about bill messages, refer to [The Source Of Bill Messages](#).

**Account Type - Controls**

You must define a variety of business rules for every division in which an account type has taxpayers. Open **Admin Menu, Account Type, Controls** tab to maintain this information.

**Description of Page**

The **Account Type Controls** scroll contains business rules governing accounts that belong to a **Division** and **Account Type**. The following fields should be defined for each **Division**:

- Use **Days Till Bill Due** to define the number of days after the bill freeze date that the taxpayer's bill is due. If the due date is a weekend or company holiday, the system will move the due date forward to the next workday (using the workday calendar defined on the account's division).

The grid that follows contains **Algorithms** that control important functions in the system. You must define the following for each algorithm:

- Specify the **System Event** with which the algorithm is associated (see the table that follows for a description of all possible events).
- Specify the **Sequence** and **Algorithm** for each system event. You can set the **Sequence** to 10 unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

**Warning!** These algorithms are typically significant system processes. The absence of an algorithm may prevent the system from operating correctly.

You can define algorithms for the following **System Events**:

System Event	Optional / Required	Description

Autopay Amount Over Limit	Optional	<p>This algorithm is called to handle the situation when a system-initiated <a href="#">automatic payment</a> is created that exceeds the taxpayer's <a href="#">maximum withdrawal limit</a>. Specifically, this algorithm is called when:</p> <ul style="list-style-type: none"> <li>- The account has a maximum withdrawal limit on their <a href="#">automatic payment options</a></li> <li>- The system attempts to create an automatic payment that exceeds this amount</li> <li>- The automatic payment algorithm that's plugged into the <a href="#">installation record</a> has logic that invokes this algorithm when the above conditions are true</li> </ul> <p>If you do not plug-in this type of algorithm and the above situation is detected, the automatic payment will be created and no error will be issued.</p> <p>Refer to <a href="#">How To Implement Maximum Withdrawal Limits</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Bill Cancel	Optional	<p>This algorithm provides the ability to include additional cancel logic when canceling online.</p> <p>Algorithms of this type can be called in two modes: D (Determine Bill Page Buttons) and X (Cancel Bill). Mode 'D' governs whether an action button to cancel the bill will appear on the Bill page and mode 'X' performs the actual cancellation logic.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Bill Completion	Optional	<p>When a bill for an account is completed, bill completion algorithms are called to do additional work.</p> <p>Refer to the description of the Complete button under <a href="#">Bill Lifecycle</a> for a description of when this algorithm is called during the completion process.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Bill Eligibility	Optional	<p>Algorithms for this plug-in spot are called when generating a bill in batch billing. It provides the ability to determine if an account is ineligible for billing and should therefore be skipped from further processing.</p> <p>If an eligibility algorithm is not used, a bill is created for any account in the open bill cycle and is later deleted by the billing process if it detects that there is no information linked to the bill.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Bill Segment Freeze / Cancel	Optional	<p>When a bill segment for an account in this account type / division is frozen or canceled, an algorithm of this type may be called to do additional work.</p> <p>Refer to <a href="#">Bill Segment Lifecycle</a> for more information about freezing and canceling bill segments.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
FT Freeze	Optional	<p>When an FT is frozen, this algorithm is called to do additional work.</p> <p>For example, if you practice <a href="#">Open Item Accounting</a>, you will need such an algorithm to handle the cancellation of match events when a financial transaction is canceled that appears on a match event. Refer to <a href="#">How Are Match Events Cancelled?</a> for more information about cancellation.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Levy an NSF Charge	Optional	<p>This algorithm is called when a payment is canceled with a cancellation reason that indicates an NSF.</p>

		<p>Refer to <a href="#">NSF Cancellations</a> for more information about what happens when a payment is canceled due to non-sufficient funds.</p> <p><b>Only One Algorithm.</b> Only one algorithm to levy an NSF charge may be defined for an account type / division combination.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Overpayment Distribution	Required	<p>When a taxpayer pays more than they owe, this algorithm is called to determine what to do with the excess funds. Refer to <a href="#">Overpayment Obligations</a> for a description on how to configure the system to handle your overpayment requirements.</p> <p><b>Only One Algorithm.</b> Only one overpayment distribution algorithm may be defined for an account type / division combination.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Override Due Date	Optional	<p>An account's bill due date will be equal to the bill date plus its account type's Days Till Due. If you need to <u>override</u> this method for accounts in a specific account type, specify the appropriate algorithm here.</p> <p><b>Only One Algorithm.</b> Only one due date override algorithm may be defined for an account type / division combination.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Payment Cancellation	Optional	<p>Algorithms of this type are called when a payment is canceled.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Payment Distribution	Required	<p>This algorithm is called to distribute a payment amongst an account's obligations. Refer to <a href="#">Payment Distribution</a> for more information about how payment distribution works.</p> <p><b>Only One Algorithm.</b> Only one payment distribution algorithm may be defined for an account type / division combination.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Payment Freeze	Optional	<p>When a payment is frozen, this algorithm is called to do additional work. If you practice <a href="#">Open Item Accounting</a>, you will need such an algorithm to link the payment's financial transactions to the match event that was originally created when the payment was distributed. Refer to <a href="#">Payments and Match Events</a> for more information.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>



Post Bill Completion	Optional	<p>When an account type has algorithms of this type, they are called after the completion of a bill for an account linked to this account type.</p> <p>Refer to the description of the Complete button under <a href="#">Bill Lifecycle</a> for a description of when this algorithm is called during the completion process.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
Pre Bill Completion	Optional	<p>When an account type has algorithms of this type, they are called immediately before completion starts for an account linked to this account type. These algorithms have the potential of:</p> <ul style="list-style-type: none"> <li>Deleting a bill. You might want a pre completion algorithm to delete a bill if a condition is detected that should inhibit the sending of a bill to a taxpayer (e.g., the bill just contains information about recent payments).</li> <li>Aborting the completion process and creating a bill exception. If the algorithm indicates this should be done, the bill is left in the <i>pending</i> state and a bill exception is created describing why completion was aborted. You might want a pre completion algorithm to do this if, for example, integrity checks detect there is something wrong with the account or its obligations. If the integrity check fails, the bill can be left in the <i>pending</i> state and a bill exception created describing why.</li> </ul> <p>Refer to the description of the Complete button under <a href="#">Bill Lifecycle</a> for a description of when this algorithm is called during the completion process.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>

## Setting Up Collection Classes

Refer to [Setting Up Collection Classes](#) for a description of how to set up collection classes.

## Setting Up Customer Contact Options

This section describes tables that must be set up before you can define customer contacts.

Refer to [The Big Picture Of Customer Contacts](#) for more information about taxpayer (customer) contacts.

### Contents

- [Setting Up Letter Templates](#)
- [Setting Up Customer Contact Classes](#)
- [Setting Up Customer Contact Types](#)

## Setting Up Letter Templates

You can set up a customer contact type to generate a form letter whenever a customer contact of this type is added. In fact, this is the only way to generate a letter in the system.

Refer to [Printing Letters](#) for more information about how letters are produced.

Every customer contact that causes a letter to be sent must reference a unique letter template. To define a letter template, open **Admin Menu, Letter Template**.

### Description of Page

The following fields are required for each letter template:

- **Letter Template** is the unique identifier of the letter template.
- Use **Description** to enter a brief description of the letter.
- Use a **Customer Contact Business Object** to define a [BO](#) that may govern additional rules related to customer contacts of this type.
- Turn on **Special Extract** if this type of letter should only be created via a system generated event such as a collection letter. Turning on this switch is what prevents a user from adding a customer contact that references this type of letter template (because you don't want a user to be able to request a letter associated with a system generate event by adding a customer contact, rather, they must execute the appropriate process and it will generate the customer contact).
- The next two fields control how letters of this type are printed (both in batch and online). Refer to [Technical Implementation Of Batch Letter Production](#) for more information about producing letters in batch. Refer to [Technical Implementation Of Online Letter Production](#) for more information about online letter production.
  - Use **Batch Control** to define the process that creates the flat file that is passed to your letter printing software. If you use an **Extract Algorithm** to construct the downloaded information, you can use the **LTRPT** process.
  - Use **Extract Algorithm** to define the plug-in component that constructs the "flat file records" that contain the information to be merged onto letters of this type. This algorithm is called when a user requests an online image of a letter on [Customer Contact - Main](#) and it may also be called by the batch letter extraction process defined above. Click [here](#) to see the algorithm types available for this plug-in spot.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI LETTER TMPL](#).

## Setting Up Customer Contact Classes

Every customer contact record has a contact type that classifies the record for reporting purposes. And every contact type, in turn, references a customer contact "class". The class categorizes customer contacts into larger groupings for reporting purposes.

Open **Admin Menu, Customer Contact Class** to define your customer contact classes.

### Description of Page

Enter a unique **Contact Class** and **Description** for each customer contact class.

After you have created your customer contact classes, you'll be ready to setup your [customer contact types](#).

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_CC\\_CL](#).

## Setting Up Customer Contact Types

Every customer contact record has a contact type that controls the behavior of the customer contact.

Refer to [The Big Picture Of Customer Contacts](#) for more information about customer contacts.

Open **Admin Menu, Customer Contact Type** to define your customer contact types.

### Description of Page

Every customer contact type is identified by a unique combination of **Contact Class** and **Contact Type**.

Enter a brief **Description** of the customer contact type.

Only specify a **Contact Shorthand** if customer contacts of this type can be added in the [Customer Contact Zone](#). The value you specify in this field is what the user selects to add a customer contact in this zone.

Use **Contact Action** if something should be triggered when customer contacts of this type are added. The only valid value in this release is **Send Letter**. If you select this option, you must also specify a **Letter Template**. Refer to [Printing Letters](#) for more information about how letters are produced.

Use the **Customer Contact Type Characteristics** collection to define characteristics that can be defined for contacts of a given type. Use **Sequence** to control the order in which characteristics are defaulted. Turn on the **Required** switch if the **Characteristic Type** must be defined on customer contacts of a given type. Turn on the **Default** switch to default the **Characteristic Type** when customer contacts of the given type are created. Enter a **Characteristic Value** to use as the default for a given **Characteristic Type** when the **Default** switch is turned on.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_CC\\_TYPE](#).

## Setting Up Filing Calendars

Filing calendars are used to define the expected filing period dates for a given type of obligation. The valid filing calendar that governs an obligation's filing period is typically defined on the obligation's [Tax Role](#) or its [Tax Type](#).

To set up a filing calendar and its filing periods, open **Admin Menu, Filing Calendar**.

### Description of Page

Enter a unique **Filing Calendar** and **Description** for the calendar.

Use **Calendar Type** to categorize your calendar. For example, you may use calendar type to define the frequency like Monthly or Quarterly. You may also choose to use calendar type to distinguish special calendars like fiscal calendars. No base values are provided for this field.

**Note.** Define values for this field using the Lookup table. This field name is FILING\_CAL\_TYPE\_FLG.

Specify the **Begin Date**, **End Date**, **Due Date** and **Grace Date** for each filing period. Refer to [Due Dates for Filing Period Based Obligations](#) for more information about due dates.

As time passes, you will need to return to this transaction to manually enter ensuing years. You can enter several years at a time or incorporate the task into end-of-year system maintenance.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_FILING\\_CAL](#).

## Setting Up Tax Types

The tax type is used to defined business rules for each type of tax you support. Tax type is required when defining any financial obligation so a catchall tax type of **Other** for write-offs, payment plans and other miscellaneous uses may be warranted.

A tax type is governed by a [business object](#). The base product provides business objects that you may use or extend. Refer to the BO definitions in the application. Or you may create your own.

Open **Admin Menu**, **Tax Type** to define the tax types used to define information about the tax types you support.

The topics in this section describe the base-package zones that appear on the tax type portal.

### Contents

- [Tax Type List](#)
- [Tax Type Actions](#)
- [Tax Type](#)

## Tax Type List

The Tax Type [List zone](#) lists every tax type. The following functions are available:

- Click the [broadcast](#) icon to open other zones that contain more information about the adjacent tax type.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each tax type.

Click the **Add** link in the zone's title bar to add a new tax type.

## Tax Type Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions are available.

## Tax Type

The Tax Type zone contains display-only information about a tax type. This zone appears when a tax type has been broadcast from the Tax Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI SVC TYPE](#).

## Setting Up Obligation Options

---

This section describes tables that must be set up before you can define obligations.

### Contents

- [Setting Up Industry Codes](#)
- [Setting Up Tax Exempt Types](#)
- [Setting Up Contract Quantity Types](#)
- [Obligation Type Controls Everything](#)
- [Financial Controls](#)

## Setting Up Industry Codes

An obligation can reference an industry code. This code is used to categorize obligations for reporting purposes. To define an industry code, open **Admin Menu, Industry Code**.

### Description of Page

Enter a unique **Industry Code** and **Description**.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI SIC](#).

## Setting Up Tax Exempt Types

Your rates will probably have provisions for the calculation of taxes of one type or another. Frequently you will have customers who are completely or partially exempt from these taxes. The obligations for these customers will need to have tax exemption information in order for them to be billed properly. Tax Exempt Type is used to define the precise nature of the applicable exemption. To define the Tax Exempt Types you will use, open **Admin Menu, Tax Exempt Type**.

### Description of Page

Enter a unique **Tax Exempt Type** and **Description** for each type of tax exemption.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_TAX\\_EX\\_TYPE](#).

## Setting Up Contract Quantity Types

You may have customers whose contracts (obligations) have contractual limits. For example, a vehicle tax account may have special considerations or limits on the amount it can be billed, such as vehicles for government agencies or diplomats. The obligations for these customers must have information regarding this quantity in order to be billed properly. Contract Quantity Type is used to precisely define the nature of the quantity. To define the Contract Quantity Types, open **Admin Menu, Contract Quantity Type**.

**Description of Page**

Enter a unique **Contract Quantity Type** and **Description** for each type of contract quantity.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_CONT\\_QTY\\_TYP](#).

## Obligation Type Controls Everything

Every obligation references an obligation type. The obligation type controls all aspects of an obligation's behavior including how bills are created, how its financial transactions are booked in the general ledger, and much more. We don't explain how to set up obligation types in this section because it's only after you have set up all of the control tables in this manual that you'll be able to finally define your obligation types.

For more information about obligation types, refer to <a href="#">Defining Obligation Types</a> .
---

## Financial Controls

There are also a number of control tables that must be set up to control the bills, payments, and adjustments that are linked to an obligation. For more information about these tables, please refer to [Defining Financial Transaction Options](#).

# Defining Property Options

This chapter describes options related to physical locations and to property owned by a taxpayer. A location is the physical location associated with an asset, such as a vehicle or building, that a taxpayer pays taxes on. The following sections describe the control tables needed to support this functionality.

## Contents

- [Setting Up Location Options](#)
- [Setting Up Location Postal Defaults](#)

## Setting Up Location Options

---

This section describes tables that must be set up before you can define locations.

## Defining Location Types

Open **Admin Menu, Location Type** to define the location types used to categorize your locations.

### Description of Page

Enter a unique **Location Type** and a **Description** for every location type.

Use a **Location Business Object** to define a [BO](#) that may govern additional rules related to locations of this type.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_PREM\\_TYPE](#).

## Setting Up Location Postal Defaults

---

You set up postal defaults if your revenue authority is able to default field values onto new locations based on the location's postal code. The topics in this section describe how to maintain postal defaults.

For more information about where these default values are used, refer to [Maintaining Locations](#).

## Postal Defaults - Main

To define location postal defaults, open **Admin Menu, Postal Code Default** and navigate to the **Main** tab.

### Description of Page

Enter the **Country Code** and range of postal codes to which the default values apply using the **From Postal Code** and **To Postal Code**.

**Note.** You may not have postal defaults whose from / to postal codes overlap.

Enter the **County** to be defaulted onto new locations located in this postal code range.

Enter the **City** to be defaulted onto new locations located in this postal code range.

Enter the **Division** to be defaulted onto new locations located in this postal code range.

Enter the **State** to be defaulted onto new locations located in this postal code range.

Enter the **Time Zone** to be defaulted onto new locations located in this postal code range.

Use the **Characteristic Types and Values** collection to define the **Characteristic Types** and their respective **Characteristic Values** to be defaulted on locations located in this postal code range. In addition to providing interesting information, these characteristics may also determine the prices and rates on the bills generated for properties associated with this location.

For more information about characteristics, see [Setting Up Characteristic Types & Their Values](#) and [An Illustration Of A Rate Factor And Its Characteristics](#).

Use the **Geographic Types and Values** collection to define the **Geographic Types** and their respective **Values** to be defaulted on locations located in this postal code range. In addition to providing interesting information, these values may be used to sort field activities in geographic value order.

#### **Where Used**

This information is defaulted when a new location is added. Characteristics and geographic values are also defaulted when the postal code for a location is changed. Refer to [Maintaining Locations](#) for more information.



# Defining Forms Processing Options

Forms processing is a core function for a tax authority.

Tax forms are the most common types of forms. These forms are processed in the context of a filing period. When a tax form is filed, it satisfies an obligation to file and its processing results in an assessment.

Registration forms are used for taxpayer registration and/or taxpayer demographic information updates. These forms are processed in the context of a taxpayer. When a registration form is processed, it usually results in the creation of new taxpayers, new accounts and / or new tax roles or updates to an existing taxpayer's information - e.g. name / address change.

Since forms processing volumes tend to be high, forms are usually processed in batch - be it via upload or manual data entry.

## Contents

- [The Big Picture of Registration](#)
- [The Big Picture of Tax Forms](#)
- [Setting Up Form Processing Options](#)
- [The Big Picture of Form Batch Processing](#)

## The Big Picture of Registration

---

A taxpayer is registered in any of the following ways:

- The taxpayer files a paper registration form
- The taxpayer submits a registration form online.
- A tax authority user registers the taxpayer using an online registration form
- For some tax types, the system can automatically register the taxpayer when the first tax form is processed. See [Automatic Taxpayer Registration](#) for more details.

## Contents

- [Registration Forms Have A Common Lifecycle](#)
- [Registration Form Processing Rule Categories](#)
- [Registration Processing Rules Are Algorithms](#)
- [Stopping Registration Form Processing When There Are Issues](#)
- [Registration Form Type Controls Form Processing](#)
- [Designing Your Registration Form Types](#)
- [Business Registration Form Type Example](#)
- [Business Registration Form Example](#)

## Registration Forms Have A Common Lifecycle

Most registration forms go through a similar lifecycle:

- The form is created in an initial state where processing rules are not yet executed. This allows the form to be saved as work-in-progress. A form in this state can be canceled.

- A validation step verifies the information on the registration form and tries to identify the taxpayer. Any issues detected from this step causes form processing to stop until the issues are resolved. Some issues may also trigger correspondence to the taxpayer.
- When the form passes validation, it is posted in the system. Actions include creating the taxpayer / account / tax role (if it does not already exist), creating customer contacts to confirm registration, etc.

**The base package provides a sample registration form *C1-BusinessRegistrationForm*.**  
Refer to the Business Object metadata for an example of a common registration form lifecycle.

## Registration Form Processing Rule Categories

Specific processing rules are executed at key points in the lifecycle of a registration form. Each tax authority will have a unique set of rules. And one set of processing rules could be more complex than others. The following sections describe broad categories of registration form processing rules.

### Contents

- [Registration - Taxpayer Existence](#)
- [Registration - Validation](#)
- [Registration - Update](#)
- [Registration - Correspondence Triggers](#)

### Registration - Taxpayer Existence

Taxpayer existence rules take key demographic information (e.g. taxpayer's name and unique identifier) from the registration form and match the information to an existing taxpayer. These rules may also include checks for existing accounts and tax roles (if tax roles are applicable to the tax type).

### Registration - Validation

Validation rules determine if the information supplied on the registration form can be processed. This usually includes checking that required information is supplied and validating that the data is in the correct formats.

### Registration - Update

Update rules execute when the registration form posts in the system. If prior taxpayer existence rules did not find an existing taxpayer, the taxpayer / account / tax role(s) are created. If an existing taxpayer was found, update rules can include creation of new accounts and tax role(s). In addition, an existing taxpayer's demographic information may also be updated.

### Registration - Correspondence Triggers

A registration form may trigger letters to be sent to the taxpayers, for example, requesting to verify information provided on the form or to provide missing information.

## Registration Processing Rules Are Algorithms

Because processing rules can be unique to each tax authority, they are implemented as algorithms.

**The base package provides a sample registration form *C1-BusinessRegistrationForm*.** Refer to the Business Object metadata for details on the algorithms provided with this business object.

## Stopping Registration Form Processing When There Are Issues

A registration form can suspend if there are errors that a tax authority user needs to review or fix.

If the registration form is missing key information, the form can go into a 'waiting' state until the tax authority receives the information from the taxpayer.

## Registration Form Type Controls Form Processing

Each registration form must reference a form type. The form type may reference business objects that define:

- The structure, lifecycle, processing rules and processing options of the form type itself. The structure of the form type includes parameters used in registration form processing.
- The structure, lifecycle, processing rules and processing options of the actual registration form.

## Designing Your Registration Form Types

The following points describe a recommended design process:

- Examine the lifecycles of your registration forms. If the lifecycles are similar, consider configuring the lifecycle in a parent business object.
- Examine the structure of your registration forms. Identify common elements that can be defined in either a parent business object's schema or data areas.
- Design your form processing algorithms. If there are processing rules that apply across different forms, consider plugging in the processing algorithms on a parent business object.
- Configure your 'business' Person Types. Define the primary identifier type (for example, EIN, SSN, and so on). Define the valid relationships between the business and the responsible parties of the business. Refer to [Defining Person Types](#) for more details.
- Configure your Tax Types. Specify whether or not Tax Role is required. Define valid calendars (if applicable) and designate a default calendar. Refer to [Setting Up Tax Types](#) for more details.
- Identify parameters for forms processing and configure them on the form type. These parameters may include the following:

- Account Types / Obligation Types. For tax forms that can automatically register the taxpayer, configure the appropriate account types and obligation types. Ensure that the account types define the appropriate collection classes and divisions. Refer to [Setting Up Account Types](#) and [Defining Obligation Types](#) for more details.
- To Do Types. Configure To Do Types for 'suspense' and 'wait timeout' To Do Entries. Refer to [Defining To Do Types](#) for more details.
- Customer Contact Classes / Types. Configure Customer Contact Classes / Types for any correspondence triggered from the tax form, for example, a letter requesting the taxpayer to provide missing information. Refer to [Setting Up Customer Contact Classes](#) and [Setting Up Customer Contact Types](#) for more details.
- Relate each form type to a specific registration form (transaction) business object.

## Business Registration Form Type Example

The base package provides a sample business registration form type business object **C1-BusRegistrationFormType**. This defines the parameters for processing the sample business registration form **C1-BusinessRegistrationForm**.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle, processing rules and processing options.

## Business Registration Form Example

The base package provides a sample business registration form business object **C1-BusinessRegistrationForm**. This is a simple example of a registration form for business tax types like Sales & Use. Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

### Contents

[Ownership Type Controls Taxpayer Creation](#)  
[Owners / Responsible Parties Are Persons](#)

### Ownership Type Controls Taxpayer Creation

This business registration form specifies an ownership type, which indicates the legal form of business, such as a corporation, partnership, sole proprietorship, etc.

The business taxpayer is created using the ownership type specified on the form.

**Ownership Type is Person Type.** Refer to [Defining Person Types](#) for more details.

**C1-BusinessRegistrationForm** assumes that the ownership types define a primary identifier type (e.g., employer identification number, social security number, etc). This is used as the business taxpayer's identifier type. Refer to [Defining Identifier Types](#) for more details.

The owners / responsible parties are linked to the business via person-to-person relationships. **C1-BusinessRegistrationForm** assumes that the ownership types define a default person relationship type. Refer to [Defining Person Relationship Types](#) for more details.

Refer to the Business Object metadata for details on the taxpayer creation logic.

## Owners / Responsible Parties Are Persons

**C1-BusinessRegistrationForm** assumes that the owners / responsible parties are persons/individuals.

Refer to the Business Object metadata for details on the processing of owners / responsible parties.

# The Big Picture of Tax Forms

---

A tax form fulfills an obligation to file a tax return for a specific filing period. Tax form processing results in an assessment - which determines either a tax amount due or an overpayment.

## Contents

- Tax Forms Have A Common Lifecycle
- Tax Form Processing Rule Categories
- Tax Form Processing Rules Are Algorithms
- Stopping Tax Form Processing When There Are Issues
- Automatic Taxpayer Registration
- Payments Related to Tax Forms
- Tax Forms Create Adjustments
- Tax Forms Can Create Overpayment Processes
- Changes After A Tax Form Posts
- Tax Form Type Controls Form Processing
- Designing Your Tax Form Types
- Tax Form Type Examples
- Tax Form Examples

## Tax Forms Have A Common Lifecycle

Most tax forms go through a similar lifecycle:

- The form is created in an initial state where processing rules are not yet executed. This allows the form to be saved as work-in-progress. A form in this state can be canceled.
- A validation step verifies the information on the tax form and tries to identify the taxpayer. Any issues detected from this step causes form processing to stop - until the issues are resolved. Some issues may also trigger correspondence to the taxpayer.
- When the form passes validation, it is posted in the system. Actions include creating the taxpayer / account / tax role (if non existing), creating financial transactions, etc.
- A posted form can be adjusted, transferred or reversed.

**The base package provides a sample tax form *C1-TaxForm*.** Refer to the Business Object metadata for an example of a common tax form lifecycle.

## Tax Form Processing Rule Categories

Specific processing rules are executed at key points in a tax form's lifecycle. Each tax authority will have a unique set of rules. And one set of processing rules could be more complex than others. Tax form processing can involve complex rules. The following are broad categories of these processing rules.

### Contents

- Tax Form - Taxpayer Existence
- Tax Form - Pre-Population
- Tax Form - Validation
- Tax Form - Calculation
- Tax Form - Data Matching
- Tax Form - Compliance
- Tax Form - Risk Based Thresholds and Overrides
- Tax Form - Update
- Tax Form - Correspondence Triggers
- Tax Form - Interface Triggers

### Tax Form - Taxpayer Existence

Taxpayer existence rules take key demographic information (e.g. taxpayer's name and unique identifier) from the tax form and match the information to an existing taxpayer. These rules may also include checks for existing accounts and tax roles (if tax roles are applicable to the tax type).

### Tax Form - Pre-Population

Pre-population rules fill in information on a tax form prior to data entry

### Tax Form - Validation

Validation rules determine if the information supplied on the tax form can be processed. This usually includes checking that required information is supplied and validating that the data is in the correct formats.

### Tax Form - Calculation

Calculation rules check that the computations on the tax form are correct.

### Tax Form - Data Matching

Data matching rules validate specific information on the tax form against data that resides on the account / obligation or other tax forms.

### Tax Form - Compliance

Compliance rules check for "red flags" on the tax form. When compliance rules are not met, it causes form processing to stop temporarily - for tax authority review. For example, a rule could state that the authority wants to review an income tax form when the total deductions are more than 45% of the adjusted gross income.

## Tax Form - Risk Based Thresholds and Overrides

These rules use pre-calculated values from a business intelligence process as processing thresholds. For example, a BI process may pre-calculate the expected minimum gross receipts amounts for every sales taxpayer. The calculated amount is used to flag tax returns that are below this threshold. A tax authority may specify an override value for this threshold.

## Tax Form - Update

Update rules execute when the tax form posts in the system. If prior taxpayer existence rules did not find an existing taxpayer, the taxpayer / account / tax role(s) are created. If an existing taxpayer was found, update rules can include creation of new accounts and tax role(s). In addition, an existing taxpayer's demographic information may also be updated.

This also includes creation of financial transactions such as the assessment and any applicable tax credits.

## Tax Form - Correspondence Triggers

A tax form may trigger a letter to be sent to a taxpayer requesting to verify information provided on the form or to supply missing information.

## Tax Form - Interface Triggers

Some tax forms may trigger an interface. For example, an income tax return may claim a credit that needs to be verified by another government agency.

## Tax Form Processing Rules Are Algorithms

Because processing rules can be unique to each tax authority, they are implemented as algorithms.

The base package provides three tax form business objects - ***C1-TaxForm***, ***C1-IndIncomeEZTaxForm*** and ***C1-SalesAndUseTaxForm***.

Refer to the Business Object metadata for details on the processing rules provided with ***C1-TaxForm***, ***C1-IndIncomeEZTaxForm*** and ***C1-SalesAndUseTaxForm***.

## Stopping Tax Form Processing When There Are Issues

A tax form can suspend if there are errors that a tax authority user needs to review or fix.

If the tax form is missing key information, the form can go into a 'waiting' state until the tax authority receives the information from the taxpayer.

## Automatic Taxpayer Registration

For tax types that do not require a taxpayer to preregister with the tax authority, the taxpayer can be automatically registered when the first tax form is processed.

The ***C1-IndIncomeEZTaxForm*** business object includes auto-registration capability. Refer to the following algorithm types for more details:

- [C1-TXTPEX](#) - Creates the taxpayer
- [C1-CRETXAR](#) - Creates the spouse (if any), account and tax role (if applicable)
- [C1-CREINDOB](#) - Creates the obligation for the filing period

The **C1-SalesAndUseTaxForm** business object includes an ability to automatically create the obligation for the filing period. Refer to algorithm type [C1-SUT-CREOB](#) for more details.

**C1-SalesAndUseTaxForm** assumes taxpayer, account and tax role pre-exist - i.e. created during business registration.

## Payments Related to Tax Forms

The following topics describe processing of payments related to tax forms.

### Contents

- [Payments In Advance / Estimated Payments](#)
- [Payments With A Tax Form](#)
- [Pay A Form / Filing Period Algorithm - Example](#)
- [Find / Transfer Payments Algorithm - Example](#)

### Payments In Advance / Estimated Payments

Some tax types require taxpayers to make estimated payments based on specific conditions -e.g. estimated revenue, estimated tax liability, etc.

When the payment is processed, the obligation may or may not exist yet.

- If the obligation exists, the payment is applied to that obligation
- If an obligation does not exist but the taxpayer has an existing account, the payment is applied to an excess credit obligation under that account.
- If the taxpayer does not have any accounts, the payment is applied to a company suspense account.

In the cases where the payment got applied to an excess credit obligation or company suspense, it's important for the payment to capture details that will facilitate the transfer of the payment to the correct obligation when the tax form gets processed. The payment should capture details like the taxpayer's identifier and the tax type / filing period being paid.

### Payments With A Tax Form

When payments are sent with a tax form, the payment is usually linked to the form by a unique identifier, such as a document locator. The payment is applied depending on which of the payment and the tax form gets processed first:

- If the tax form is processed before the payment, the payment is applied to the obligation on the form.
- If the payment is processed before the tax form and:
  - The taxpayer has an existing account, the payment is applied to an excess credit obligation under that account



- The taxpayer does not have any accounts; the payment is applied to a company suspense account.

In the cases where the payment got applied to an excess credit obligation or company suspense, it's important for the payment to capture details that will facilitate the transfer of the payment to the correct obligation when the tax form gets processed. The payment should capture details like the document locator, taxpayer's identifier and the tax type / filing period being paid.

### Pay A Form / Filing Period Algorithm - Example

The base package provides a **Pay A Form / Filing Period** Distribution Rule - Create Payment algorithm type [C1-PAYFRM](#). This algorithm type assumes that the payment event distribution details can be specified as any of the following

- Document locator only - Algorithm will post the payment to the tax form's obligation. If the tax form is not found, the payment is applied to either an excess credit obligation or company suspense account.
- Document locator with taxpayer identifier number, tax type and filing Period - Algorithm will post the payment to the tax form's obligation. If the tax form is not found, the payment is applied to either an excess credit obligation or company suspense account.
- Taxpayer identifier number, tax type and filing period (i.e. advanced / estimated payments) - Algorithm will post the payment to either an excess credit obligation or to company suspense account.

To enable this algorithm type:

- Configure an algorithm for C1-PAYFRM with the appropriate parameters. Use base characteristic types for DLN, Taxpayer Identifier, Tax Type and Filing Period.
- Create distribution rules for each of the distribution details. Use base characteristic types for DLN, Taxpayer Identifier, Tax Type and Filing Period.
- Plug in your Pay A Form / Filing Period algorithm in one of the distribution rules you created.

**The distribution detail with the 'create payment' algorithm is last.** When creating the payment event using the above distribution rules, the detail that has the 'create payment' algorithm must be specified as the last. This ensures that all details are available by the time the algorithm executes.

### Find / Transfer Payments Algorithm - Example

The base package provides a **Find / Transfer Payments** algorithm type [C1-TRFPAY](#) that can be plugged on the Posted state of the tax form business objects.

This algorithm type is designed to work with the **Pay A Form / Filing Period** Distribution Rule - Create Payment algorithm type [C1-PAYFRM](#).

It looks for any payments that may have posted prior to tax form processing - using the same distribution details that C1-PAYFRM populated on the payments.

To enable the Find / Transfer Payments algorithm type:

- Configure an algorithm for C1-PAYFRM with the appropriate parameters. Use the same char types used in your distribution rules and in your Pay A Form / Filing Period algorithm.

- Plug in your Find / Transfer Payments algorithm on the 'posted' state of your tax forms business objects

**Obligation must exist.** The Find / Transfer Payment algorithm expects that the obligation has been identified. Thus, your algorithm must be plugged in after obligation existence / creation logic.

## Tax Forms Create Adjustments

Most tax forms create an assessment for the obligation. Assessments are created as Adjustments. In addition, other adjustments can be created for specific tax credits that are applicable to the tax form.

### Penalty and Interest

If [penalty and interest](#) (P&I) rules are applicable to the obligation associated with the tax form, the various adjustments created when posting the tax form may impact penalty and interest calculations.

As described in [Adjustments and Updating P&I](#), the base product does not recommend configuring adjustments level plug-ins to cause P&I to be recalculated. For a business process like posting a tax form, the request to bring P&I up to date should occur after all financial effects are posted.

The base product algorithms provided with the base tax form business objects call the process to recalculate P&I after all forms are posted.

## Tax Forms Can Create Overpayment Processes

Tax form processing can initiate overpayment processing if the obligation ends up being overpaid after the tax form is processed.

The ***C1-IndIncomeEZTaxForm*** base business object includes this ability to create overpayment processes.

Refer to [The Big Picture of Overpayments](#) for more details on overpayment processing.

## Changes After A Tax Form Posts

There are three actions that can be done to a tax form after it has posted in the system.

### Contents

- [Adjusting A Tax Form Online](#)
- [Transferring A Tax Form Online](#)
- [Reversing A Tax Form Online](#)
- [Standalone Amendment Forms](#)
- [Standalone Audit Forms](#)

## Adjusting A Tax Form Online

A posted tax form can be adjusted for any of the following common reasons:

- Correcting data capture errors or data entry errors
- Line item adjustments initiated by the taxpayer
- Adjustments initiated from an audit

When a tax form is adjusted, the financial effect of the existing tax form is canceled and new financial transactions are created for the updated line items.

**The base package provides a sample tax form *C1-TaxForm*.** Refer to the Business Object metadata for an example of tax from adjustment.

## Transferring A Tax Form Online

A tax form can be transferred to a different taxpayer/obligation if it previously posted to the incorrect taxpayer/obligation.

When a tax form is transferred, the financial effect of the existing tax form is removed from the current obligation and new financial transactions are created for the 'transfer to' obligation.

**The base package provides a sample tax form *C1-TaxForm*.** Refer to the Business Object metadata for an example of tax from transfer.

## Reversing A Tax Form Online

Some exceptional cases may require that the financial effect of a tax form be canceled, without necessarily creating new financial transactions. A tax form reversal is a way of voiding a tax form that has already posted.

**The base package provides a sample tax form *C1-TaxForm*.** Refer to the Business Object metadata for an example of tax from reversal.

## Standalone Amendment Forms

A posted tax form can also be adjusted by processing a standalone amendment form. In this case, the amendment form is processed independently of the original form. New financial transactions are created for the difference between the financial effect of the original form and the financial effect of the amendment form.

## Standalone Audit Forms

A standalone audit form can also adjust a previously posted tax form. In this case, the audit form is processed independently of the original form. New financial transactions are created for the difference between the financial effect of the original form and the financial effect of the audit form.

## Tax Form Type Controls Form Processing

Each tax form must reference a form type. The form type may reference business objects that define:

- The structure, lifecycle, processing rules and processing options of the form type itself. The structure of the form type includes parameters used in tax form processing.
- The structure, lifecycle, processing rules and processing options of the actual tax form.

## Designing Your Tax Form Types

The following points describe a recommended design process:

- Examine the lifecycle of your tax forms. If the lifecycles are similar, consider configuring the lifecycle in a parent business object.
- Examine the structure of your tax forms. Identify common elements that can be defined in either a parent business object's schema or data areas.
- Design your form processing algorithms. If there are processing rules that apply across different forms, consider plugging in the processing algorithms on a parent business object.
- Configure your Tax Types. Specify whether or not tax role is required. Define valid calendars (if applicable) and designate a default calendar. Indicate whether or not pre-registration is required. Refer to [Setting Up Tax Types](#) for more details.
- Identify parameters for forms processing and configure them on the form type. These parameters may include the following:
  - Account Types / Obligation Types. For tax forms that can automatically register the taxpayer, configure the appropriate account types and obligation types. Ensure that the account types define the appropriate collection classes and divisions. Refer to [Setting Up Account Types](#) and [Defining Obligation Types](#) for more details.
  - Rates (if applicable). If any of the calculation rules use rate application, configure the rate schedules (and rate factors) accordingly. Refer to [How To Create A New Rate](#) for more details.
  - Adjustment Types. Configure the different adjustment types for assessment, tax credits, and other financial transactions that are applicable to the tax form. Refer to [Setting Up Adjustment Types](#) for more details.
  - Overpayment Process Types. For tax forms that initiate overpayment processing, configure the appropriate overpayment process types. Refer to The [Big Picture of Overpayments](#) for more details.
  - To Do Types. Configure To Do Types for 'suspense' and 'wait timeout' To Do Entries. Refer to [Defining To Do Types](#) for more details.
  - Customer Contact Classes / Types. Configure Customer Contact Classes / Types for any correspondence triggered from the tax form, such as a letter requesting the taxpayer to provide missing information. Refer to [Setting Up Customer Contact Classes](#) and [Setting Up Customer Contact Types](#) for more details.
- Relate each form type to a specific tax form (transaction) business object.

## Tax Form Type Examples

The following topics describe the tax form type business objects that are included in the base package.

### Contents

- Individual Income EZ Tax Form Type
- Sales & Use Short Tax Form Type

### Individual Income EZ Tax Form Type

The base package provides a sample Individual Income EZ Tax Form Type business object **C1-IndividualIncomeEZTaxFormTy**. This defines the parameters for processing the sample individual income EZ tax form **C1-IndIncomeEZTaxForm**.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

### Sales & Use Short Tax Form Type

The base package provides a sample Sales & Use Short Tax Form Type business object **C1-SalesAndUseTaxFormTy**. This defines the parameters for processing the sample individual income EZ tax form **C1-SalesAndUseTaxForm**.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

## Tax Form Examples

The following topics describe the tax form business objects that are included in the base package.

### Contents

- Tax Form (Parent)
- Individual Income EZ Tax Form
- Sales & Use Short Tax Form

### Tax Form (Parent)

The base package provides a tax form business object **C1-TaxForm** that defines information and processing rules that are common to most tax forms.

This business object defines a typical lifecycle of a tax form. This is a parent business object to the specific tax form business objects **C1-IndIncomeEZTaxForm** and **C1-SalesAndUseTaxForm**.

Your implementation can include **C1-TaxForm** in your specific tax form business objects if the information, lifecycle, processing rules and processing options are applicable.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

### Individual Income EZ Tax Form

The base package provides a sample Individual Income EZ Tax Form business object **C1-IndIncomeEZTaxForm**.

This business object defines **C1-TaxForm** as its parent. Therefore, it inherits the information, lifecycle, and processing rules of the parent. **C1-IndIncomeEZTaxForm** defines additional information, processing algorithms and processing options that are unique to this type of form.

This sample tax form includes an ability to auto-register the taxpayer if not yet existing in the system. This form also initiates an overpayment process if form processing results in an overpayment.

The assessment amount is calculated using a rate schedule that is specified on the form type.

Penalty and interest are updated after all financial transactions are created.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

## Sales & Use Short Tax Form

The base package provides a sample Sales & Use Short Tax Form business object **C1-SalesAndUseTaxForm**.

This business object defines **C1-TaxForm** as its parent. Therefore, it inherits the elements, structure, lifecycle and processing rules of the parent. **C1-SalesAndUseTaxForm** defines additional information, processing algorithms and processing options that are unique to this type of form.

This sample tax form assumes that the business is preregistered, that is, the taxpayer, account and tax role exist. If an obligation does not exist it is created.

A financial transaction is created for the assessed amount. Penalty and interest are updated after the FT is created.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.

# Setting Up Form Processing Options

---

## Setting Up Form Types

Form Types contain the rules that control how forms are processed. To set up a Form Type, open **Admin Menu, Form Type**.

### Contents

- [Form Type Query](#)
- [Form Type Portal](#)

## Form Type Query

Use the [query portal](#) to search for an existing form type. Once a form type is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Form Type Portal

This portal appears when a form type has been selected from the Form Type Query portal.

The topics in this section describe the base-package zones that appear on this portal.

**Contents**

[Form Type Actions](#)  
[Form Type Zone](#)  
[Form Type / Child Form Type](#)  
[Form Type / Filing Period](#)  
[Form Type Log](#)

**Form Type Actions**

This is a standard [actions zone](#).

If the form type is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

**Form Type Zone**

The Form Type zone contains display-only information about selected form type.

Please see the zone's help text for information about this zone's fields.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_FORM\\_TYPE](#).

**Form Type / Child Form Type**

This zone will accept a broadcast tax form type and display the list of tax form types to which the current form type is a parent.

**Form Type / Filing Period**

This info zone will display the list of Filing Periods that are valid for the broadcasted Form Type.

**Form Type Log**

This is a standard [log zone](#).

## **The Big Picture of Form Batch Processing**

---

The following topics describe form batch processing functionality.

**Contents**

[Manual Data Entry of Forms](#)  
[Designing Form Batch Headers](#)  
[Manual Data Entry Form Batch Header Example](#)

### **Manual Data Entry of Forms**

Most large tax authorities have imaging equipment that captures paper registration and tax forms. The imaging equipment supports either optical character recognition (OCR) or 'key from image' processing. Forms are manually entered into the system for either of two common reasons:

- The imaging / capture systems fail.
- Form types are low-volume so that it is not cost-effective to build imaging profiles for them.

At smaller tax authorities, manual data entry of forms may be common.

When forms are received, they go through a variety of steps that ultimately groups similar forms into batches. Form batches contain a discrete number of forms that are usually of the same form type, form year, and disposition, that is a refund is expected, a balance due is expected, or there is an in-balance. The disposition is used to prioritize data entry work.

People who manually enter forms in the system may not necessarily be trained to resolve errors on the form. They simply key in the forms and if any of the forms fail validation, the work is routed to someone with the skill set required to address the specific issue(s). Batch controls are usually in place to ensure that the data entry worker does not skip forms or key in incorrect amounts.

When a batch of forms is created online, the batch will consist of either a specific tax form type or a specific registration form type. When batches are interfaced from a data capture system, it is possible for a batch to include a mix of form types - i.e. multiple tax form types, multiple registration form types or mixed tax / registration form types.

## Designing Form Batch Headers

If manual data entry is applicable to your implementation, the following points describe a recommended design process:

- Analyze the information you want to capture on the form batch header, such as:
  - Single form type vs. multiple form types in one batch
  - Record counts
  - Total amounts
- Examine the lifecycle of your form batch headers.
- Design your form batch header processing algorithms. Identify issues that cause form batch processing to stop until a user can resolve the issues.

## Manual Data Entry Form Batch Header Example

The base package provides a sample Form Batch Header ***C1-FormBatchHeader***. This is a form batch header for manual data entry of forms. This header groups forms of a specific form type. Processing logic includes validating that the form count matches actual number of forms in the batch. This can be extended to validate that the payment amount total match the sum of actual payment amounts specified on tax forms.

Refer to the Business Object metadata for details on the structure of the schema, lifecycle and processing rules.



# Defining Penalty and Interest Options

Penalty and Interest (P&I) is the commonly used term to describe a wide range of penalty, interest and fee liabilities. These charges are usually imposed by legislation.

- A penalty is a liability imposed for noncompliance with tax law. Revenue authorities can file penalties against taxpayers for "failure to file," "failure to pay," "substantial understatement," and so on
- Interest compensates the government for cost of money over time
- Fees are imposed to reimburse revenue authorities for their cost of doing business. For example, revenue authorities may incur fees for sending certified mail or for handling defective checks and will pass those fees onto the taxpayers

P&I may be triggered in any of the following ways:

- Event driven, for example, reversing a payment due to non-sufficient funds will impose a fee. These types of transactions are simply one-off adjustments that are created by an appropriate algorithm when the event occurs
- User triggered, for example, auditors may impose an inadequate record-keeping penalty their discretion. The amount may be predefined or determined by the auditor. These types of transactions are created manually or may be created as part of a business process defined by your implementation.
- Ongoing, system-generated charges based on predefined configuration rules. The remainder of this chapter focuses on these types of charges.

## Contents

[The Big Picture of Credit Allocation](#)  
[The Big Picture of Penalty and Interest](#)  
[Setting Up Penalty and Interest Options](#)

## The Big Picture of Credit Allocation

---

Before explaining penalty and interest logic, another important concept called credit allocation must be introduced. The following sections provide more information.

### Credit Allocation

Credit allocation is the process of taking credits within an obligation and applying business rules to allocate them against tax, penalty, interest and fees.

Credit allocation sequences specify an order to apply the credit against liability types. This is usually very specific. For example, you could set up the credit allocation sequence to allocate credits in the following order:

1. Defective check fee
2. Failure to pay penalty
3. Interest
4. Failure to pay penalty

5. Tax
6. Outsource collection agency fee

They may also differ based on the type of credit. For example,

- Withholding credit may be allocated differently than a payment
- Payment received before the due date may be allocated differently than a late payment
- Individual payments may have unique priorities dictated by a court order or overridden by a user

The following sections provide more information about credit allocation.

## Contents

[Credit Allocation Is Dynamic](#)  
[Debt Categories and their Priorities](#)  
[Determine Balance Details Algorithm](#)  
[Credit Allocation Zones](#)

## Credit Allocation Is Dynamic

The term "dynamic" is used to describe credit allocation because the system does not store the matching of credits to debits in the database. Rather, the system applies the credit allocation rules real-time when a request for the detailed balance of an obligation is performed.

## Debt Categories and their Priorities

In order to be able to apply credits to debits at the granular level shown above, each financial transaction whose amount is a debit must be assigned a debt category

- For adjustments that are debits, the debt category is defined [adjustment type](#)
- For bill segments, the debt category is defined on the [obligation type](#)
- For payments that are debits, for example "negative" payments used to refund money via direct deposit, the base product provides an FT freeze algorithm called [C1-CFTZ-RPDC](#) to plug into the Account Type that populates the debt category.

Debt categories may also be linked to credit financial transactions. The base algorithm that allocates credits to debits uses the debt category on credit financial transactions as a prioritization. See [Determine Balance Details Algorithm](#) for more information.

The debt category priority defines the order in which credits may be applied to debits.

- The default debt category priority is defined on the [obligation type](#)
- If a certain type of credit adjustment, for example a withholding credit dictates a different priority than the default, define the override debt category priority on the [adjustment type](#).
- A [financial transaction](#) could reference a debt category priority directly for unusual situations where a payment may require a special debt category priority

## Determine Balance Details Algorithm

The dynamic credit allocation is performed by an algorithm plugged into the **Determine Balance Details** plug-in spot on the [obligation type](#). The determine balance details algorithm is used throughout the base P&I calculation algorithm but may also be called by other processes to get a balance broken out by assessment / debt category.

The following information highlights the logic in the base algorithm [C1-PS-DB-STD](#).

The algorithm may receive specific FTs in which case it will only use those FTs. Otherwise, it retrieves the FTs for the obligation.

A special plug-in spot **Retrieve FT List** on [obligation type](#) is called to retrieve the FTs for an obligation.

**Base plug-in.** Click [here](#) to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

**Debt Category / Assessment on Credits.** This algorithm assumes that if a credit financial transaction references an assessment and/or a debt category that this information is used to prioritize the application of the credit, not restrict its application. This ensures that the results do not include one assessment (or debt category) with an overall debit balance and another assessment (or debt category) with an overall credit balance.

The algorithm processes credits in order from oldest to newest. For each credit found,

- If it references a debt category or assessment (group FT ID) or both, the credit is allocated first to debits with a matching debt category and / or assessment whose effective date is ON or BEFORE the credit's effective date.
- If credit is still remaining, the credit is allocated to debits whose effective date is ON or BEFORE the credit's effective date using the credit allocation rules for the appropriate [debt category priority](#) for this credit.
- If credit is still remaining, the credit is allocated to debits with a matching debt category and/or assessment whose effective date is AFTER the credit's effective date.
- If credit is still remaining, the credit is allocated to debits whose effective date is AFTER the credit's effective date using the credit allocation rules for the appropriate [debt category priority](#) for this credit.

The algorithm returns a detailed list of FTs including which credits were applied to which debits.

**FT Details.** The details of the FTs for the obligation and how the credits are allocated to the debits is not hard-coded but rather is defined in a data area supplied to the Determine Detailed Balance plug-in spot. The base algorithm uses the data area **C1-PI-MainFtInfo**. However, if your implementation requires different information to be supplied for each FT, you may introduce your own data area and an appropriate determine detailed balance algorithm to populate the details accordingly.

## Credit Allocation Zones

The base product provides zones on control central, one on the account information tab and one on the taxpayer information tab, to show the results of credit allocation for the account or taxpayer's current balance.

Refer to [Control Central - Account Information](#) for more details.

# The Big Picture of Penalty and Interest

The following sections describe topics related to penalty and interest calculations.

## Contents

- [P&I is Calculated for an Obligation's Assessments](#)
- [P&I Rules for a P&I Control Define the Calculation](#)
- [P&I Calculation Algorithm is the Engine](#)
- [Forecasting Penalty and Interest](#)
- [P&I and Cash Accounting](#)
- [When Is P&I Updated?](#)
- [Waivers](#)

## P&I is Calculated for an Obligation's Assessments

The system creates, stores, and maintains tax liability [assessments](#). These assessments may incur P&I over time. When more than one assessment exists for an obligation, the type of assessment may control P&I rates and rules. For example, if the taxpayer is being audited, harsher penalty rates and rules are used.

As described in [Credit Allocation](#) the base product determine detailed balance algorithm allocates credits for an obligation across assessments such that an obligation with multiple assessments would not incur P&I on one assessment while another assessment is in credit. As a result, the P&I calculation algorithm is invoked for an obligation and it always calculates penalty and interest for all assessments for the obligation.

## P&I Rules for a P&I Control Define the Calculation

For each obligation type where ongoing, system generated penalty and interest rules apply, you must create a P&I control with its collection of P&I Rules that define the distinct penalty, interest charge or fee.

## Contents

- [Calculating the Charge](#)
- [Designing Your P&I Control and P&I Rules](#)

## Calculating the Charge

The actual calculation of each P&I rule's charge is done in the **Rule Processing** algorithm plugged into the business object for the P&I rule. Configuration that the rule processing algorithm needs to successfully calculate the charge for each period is often captured on the P&I rule when defining it.

**Base plug-ins.** Click [here](#) to see the algorithm types available for this system event.

## Contents

[Calculation Basis](#)  
[Use Rates To Define the Penalty or Interest Rate](#)  
[Adjustment / Debt Category Configuration](#)

### Calculation Basis

In many cases the P&I charges are calculated as a percentage of outstanding debt, referred to as the calculation basis. The calculation basis may simply be the amount of unpaid tax or it may be the amount of unpaid tax plus other unpaid charges, such as unpaid penalty or unpaid interest.

The calculation basis can change over time based on changes to the legislation and recalculation of historic info should use the calculation basis in effect at the time of the charge. A change in calculation basis requires a new P&I control and a corresponding new set of rules to be created and linked to the obligation type for the correct effective date.

### Use Rates To Define the Penalty or Interest Rate

There are two ways that the actual percentage may be applied to the calculation basis: using a rate schedule or using a rate factor.

- Use a rate factor if the calculation is a simple percentage applied to the calculation basis. The rate factor can contain an effective dated list of percentage rates. Using this method, the appropriate rate factor is defined when configuring the P&I rule and the rule processing algorithm is responsible for retrieving the rate factor value and performing the calculation.
- Use a rate schedule if the calculation is more complicated and you would like to take advantage of the calculation logic built into the various types of rate components for a rate. For example, if your charge has a maximum amount that can be charged (for example, a penalty that is 2.5% of the outstanding tax up to a maximum 25% of the outstanding tax), this can be configured easily using the **Maximum** rate component type. Using this method requires the rule processing algorithm to set up all the amounts that the rate schedule requires to successfully apply the rate. For the current example, it must supply the outstanding tax amount that is the basis of calculation and the maximum amount allowed. Then it should call the rate application service. The P&I rule defines the rate schedule, rate quantity identifier (RQI) for passing in the calculation basis and any other information needed to supply to the rate application service. In this example, the Not to Exceed percentage and an RQI for passing to rates the calculated Not to Exceed amount.

**Rate Schedule vs. Rate Factor.** The simple percentage calculation may also be performed using rates. However, it requires an administrative user to configure a rate schedule, rate version and a rate component in addition to the rate factor. Designing the algorithm to apply the rate factor directly saves the need for extraneous rates configuration.

### Adjustment / Debt Category Configuration

When the penalty or interest charge is posted to the system to affect the taxpayer's balance, an adjustment is used. The adjustment type is defined on the P&I rule.

Each penalty and interest charge is identified by a debt category and this debt category is important for P&I calculation. Because adjustment type references a debt category, the P&I rules supplied with the base product do not capture the debt category explicitly. They derive the penalty or interest rule's debt category from its adjustment type.

**No Duplication of Debt Categories.** The base algorithms for calculating P&I expect that no two P&I rules for a P&I control refer to the same debt category (via its adjustment type).

## Designing Your P&I Control and P&I Rules

The base product provides a BO for P&I Control **C1-StandardPIControl**. Your implementation can add additional business rules to this BO as required. If your implementation has radically different requirements for your P&I controls, you can create a different business objects with their own business rules.

The base product supplies two BOs for P&I Rules.

- **C1-MonthlyChargePIRule** - this BO includes an algorithm that calculates the monthly charge with a Not to Exceed limit, using a rate schedule.
- **C1-SimpleCalculationPIRule** - this BO includes an algorithm that calculates a simple charge applying a rate factor to the outstanding calculation basis amount.

Your implementation can define add additional business rules to these BOs as required. If your implementation has P&I rules aren't satisfied by one of the above business objects, you may create your own using the above as samples. The following points highlight the important configuration for this business object:

- Define the appropriate rule processing algorithm to calculate the charge correctly.
- If [waivers](#) are applicable for the P&I rule's debt category, be sure to plug in appropriate rule processing algorithms to handle waiving the P&I rule's charge
- For any configuration required by the rule processing algorithm, consider whether it should be defined when configuring the P&I rule. If so, include the appropriate elements in the P&I rule business object's schema.

## P&I Calculation Algorithm is the Engine

An algorithm plugged into the **P&I Calculation** plug-in spot on the [obligation type](#) is responsible for calculating / forecasting penalty and interest for an obligation.

**Configuration Requirements.** The base product P&I calculation algorithm relies on many other algorithms plugged into obligation type for it to work properly. In addition, it expects effective P&I controls with at least on P&I rule to exist for the obligation type. Refer to [Setting Up Penalty and Interest](#) for more information.

### Contents

- [Calculation Overview](#)
- [Eligibility](#)
- [Recalculate Every Time](#)
- [Calculate For Discreet Time Periods](#)

Miscellaneous Information Needed  
Apply P&I Rules for Each Time Period  
Post Processing  
Calculation Through Date

## Calculation Overview

The following sections highlights the logic in the base algorithm [C1-PI-CALC](#). It has been designed to call many other algorithm plug-in spots to perform key steps in the calculation. The goal is for your implementation to use the base product P&I calculation algorithm and implement your organizations specific rules by plugging in appropriate algorithms in the various plug-in spots called by the base algorithm. However if the logic in the base algorithm does not satisfy your business requirements, you may introduce your own, using this algorithm as a sample.

At a high level, the base algorithm follows these steps

- Determine [eligibility](#)
- For eligible obligations
  - Determine [discreet time periods](#)
  - For each time period, apply the P&I rules
  - Post processing
- Update the obligation's calculate to date

The algorithm may be called in "update" or "forecast" mode. When calling the algorithm in "forecast" mode, the calling program may provide specific FTs in which case it will only use those FTs. Otherwise, it retrieves the FTs for the obligation.

A special plug-in spot **Retrieve FT List** on [obligation type](#) is called to retrieve the FTs for an obligation.

**Base plug-in.** Click [here](#) to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

**FT Details.** The list of the FTs for the obligation includes a lot of detail of each FT needed by the penalty and interest calculation. This list of details is not hard-coded but rather is defined in a data area supplied to the P&I Calculation plug-in spot and to the Retrieve FT List plug-in spot. The base algorithm uses the data area **C1-PI-MainFtInfo**. However, if your implementation requires different information to be supplied for each FT, you may introduce your own data area.

**Internal P&I Information.** Besides the details of the FTs passed back and forth to services that invoke this plug-in spot, the base P&I calculation algorithm uses data internally that must be passed to the various plug-in spots it invokes. The base algorithms use the data area **C1-PI-InternalCalculationInfo**. However, if your implementation requires different or additional information to be passed around to the various P&I calculation plug-in spots that support the internal P&I information, you may introduce your own data area.

## Eligibility

There are some types of taxpayers for which penalty and interest is exempt. Examples of such taxpayers include other government agencies and non-profit organizations.

The base product provides an [obligation type](#) plug-in spot **P&I Eligibility** where an implementation may plug in algorithms that check an obligation's eligibility for penalty and interest. The P&I calculation algorithm provided with the base product calls the P&I eligibility plug-ins prior to performing any penalty or interest calculations. If the algorithms indicate that the obligation is ineligible, no calculations are performed.

**Base plug-ins.** Click [here](#) to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

## Recalculate Every Time

Various events may cause the system to recalculate historic penalty and interest:

- Back-dated payment
- Waive existing penalty or interest
- Change to the obligation's override filing due date
- Canceling an assessment

For normal requests from the system to "bring P&I up to date", where historical calculations are not affected, the system still recalculates the P&I from the beginning every time.

One reason for this logic is to ensure that P&I is always accurate. It could be that something has changed in the system that DOES affect historical calculations and a request to recalculate historical P&I was not performed. The next time P&I runs, recalculating from the beginning ensures that the latest calculation reflects the correct charge.

Another important reason to recalculate P&I from the beginning every time P&I is brought up to date is to avoid rounding issues that may occur over time.

Consider the following example.

- A monthly penalty amount for a delinquent taxpayer is 34.3444
- If the system calculates, rounds and posts, each month ignoring the running total, the penalty after 2 months is
  - Month 1 calculation: 34.3444, rounded to 34.34
  - Month 2 calculation: 34.3444, rounded to 34.34
  - Total: 68.68
- If instead the system keeps track of the running total and creates the adjustment based on the latest running total, the calculation is as follows:
  - Month 1 calculation: 34.3444, rounded to 34.34
  - Month 2 calculation:  $34.3444 + 34.3444 = 68.6888$  less amount already posted (34.34): 34.3488, rounded to 34.35
  - Total: 68.69



Over time these rounding differences add up. To avoid that the system calculates from the beginning every time P&I is calculated and the running total is kept throughout the calculations.

### Calculate For Discreet Time Periods

There are many discreet time periods for which P&I must be calculated:

- Every date for which a credit financial transaction exists (because the calculations are based on an outstanding balance, which is affected by each credit).
- Any effective dated change in the P&I Control linked to the obligation
- Any accrual day of the month. This is especially necessary if you have a monthly charge that should accrue on a given day of the month and the charge includes other charges in its calculation basis. For example, if penalty accrues on the first day of the month and includes interest in its calculation basis, you must stop and calculate interest and then the penalty on the accrual day of the month so that your calculation basis is accurate.
- More... Your implementation may identify other events in the system that affect P&I calculations. For example, if a bankruptcy is logged in the system as a case, P&I should be calculated up to the bankruptcy start date and then skipped until the bankruptcy end date

The system provides an [obligation type](#) plug-in spot **P&I pre-processing** to use for building up the collection of dates that affect P&I calculations. This plug-in spot may also be used for retrieving any other information needed during the P&I calculation.

**Base plug-ins.** Click [here](#) to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

### Miscellaneous Information Needed

Besides determining the [time periods](#) for with P&I must be calculated, **P&I pre-processing** plug-ins on [obligation type](#) may be used to retrieve any other information needed during the P&I calculation.

For example, the base product provides a plug-in to retrieve details about waivers that exist for any of the obligation's assessments. This information is used by the rules processing algorithms.

### Apply P&I Rules for Each Time Period

Once the discreet time periods for P&I calculation are determined, the base P&I calculation algorithm needs to determine the balance by debt category for the time period in question and call the P&I rule algorithms.

To calculate the balance by debt category, [Determine Balance Details](#) algorithm is used. However, because the P&I calculation algorithm is recalculating P&I from the beginning every time and needs the balance recalculated for each period, it needs to be explicit about which financial transactions are used in the balance calculation. For example:

- When calculating the balance as of a given date, credits effective on that date should be ignored because the system is trying to find the balance as of that date prior to applying the credit. However all debits on or before the given date should be considered.
- Existing P&I transactions should not be factored into the calculation. Only the ones created by this P&I run

- Existing [waiver](#) transactions should not be factored into the calculation. However, waiver transactions corresponding to the P&I transactions calculated in the current run must be considered.

To ensure that the appropriate financial transactions are used to calculate the balance for each time period, a special plug-in spot on the [obligation type](#) is used: **P&I Prepare Periodic Balance**. This plug-in is responsible for calling the **Determine Detailed Balance** plug-in passing the appropriate financial transactions. Note that this plug-in spot receives an indication as to whether it's the **Initial**, **Interim** or **Final** call to get the balance details. This information allows the plug-in to do extra steps at the beginning or at the end.

**Base plug-ins.** Click [here](#) to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

Once the balance is available for each time period, for each assessment linked to the obligation, the P&I calculation algorithm invokes the rule processing algorithm for each P&I rule in effect for the obligation type's P&I control during this time period.

**No updates.** The rules processing algorithms provided in the base product do not do any updates to the database for new charges or cancellation of existing charges. These algorithms simply update the internal P&I information with new charges calculated and indicating any charge that should be canceled. Post processing algorithms are responsible for creating / cancelling financial transactions.

After all time periods are processed, the balance details are calculated one more time to produce the final results.

## Post Processing

The system provides an [obligation type](#) plug-in spot **P&I post-processing** to perform any steps that need to occur at the end of all the P&I calculations.

In the base P&I calculations, post processing algorithms are used to compare the calculated penalty and interest charges and calculated waiver amounts against the existing financial transactions in the system. The algorithms create or cancel financial transactions to match the latest calculations. This is only applicable when the P&I algorithm is called in **update** mode.

**Base plug-ins.** Click [here](#) to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

## Calculation Through Date

When penalty and interest is updated for an obligation, it is updated with a Calculation Through Date. This information is captured so that users reviewing the detailed balance for an obligation knows how recently the P&I calculation has occurred.

## Forecasting Penalty and Interest

Your implementation may include business processes that require forecasting of an obligation's balance to a current or future date without posting the P&I transactions:

- Sending a bill may include amount to pay if late, for example:
  - If you pay by January 1, please pay \$2000
  - If you pay by January 15, please pay \$2020
  - Etc,
- Sending a collection notice may include "please pay X amount by Y date" where the X amount is the forecasted balance on that date, which P&I included.
- If a payment is received at an account or taxpayer level where the taxpayer has several unpaid obligations, the payment algorithm that determines how to distribute the amount across obligations. When determining how much to direct to each obligation, the algorithm should forecast P&I to the payment event's effective date so that an accurate picture of the balance of each obligation is known.
- [Proposing scheduled payments](#) for a pay plan should include logic to forecast P&I to the future so that the scheduled payments cover P&I charges that will continue to accrue

The P&I calculation algorithm may be called to forecast. When forecasting, the algorithm perform all the same calculations, but does not store or cancel any adjustments to affect the obligation's balance.

A service that calls P&I calculation in forecast mode may optionally provide financial transactions to use. If financial transactions are not provided, the algorithm retrieves the FTs currently linked to the obligation and forecasts P&I to the input date. If FTs are provided, the algorithm uses them in the P&I calculations. Supplying FTs allows a calling program to forecast the P&I with a "what if" scenario. For example "what if the taxpayer makes a payment on date X?"; this is useful when proposing scheduled payments for a payment plan.

## P&I and Cash Accounting

It is common for tax authority to practice cash accounting for certain situations. For example, perhaps your implementation is a state revenue agency collects sales tax revenue on behalf of counties in your state. You may only distribute the revenue to the counties after the payment is made and not at the time of posting the assessment.

Refer to [Payables Cash Accounting](#) for general information about cash accounting functionality.

When dynamic credit allocation is practiced, where amounts are directed specifically to tax, penalty, interest and fees in a specific order, the allocation of credits to the various debt categories must be performed before any updates to the general ledger can be made as a result of cash accounting.

The base product provides a **P&I post processing** algorithm ([C1-PI-PS-CSH](#)) to make all necessary adjustments to the general ledger to transfer amounts from "holding" general ledger accounts to true payable accounts. Refer to the algorithm type description for the details.

## When Is P&I Updated?

There is no hard coded logic in the system to invoke the penalty and interest calculation. All requests to bring penalty and interest up to date are executed using plug-in logic. The base product provides logic to forecast P&I or bring P&I up to date when certain events occur. Your implementation may introduce new events that should cause P&I to be recalculated.

The following events in the system cause P&I to be brought up to date

- When a [tax form](#) posts or is reversed, transferred or adjusted. Basically any state transition of a form that causes adjustments to be created should include a step to bring P&I up to date.
- When a payment is frozen or canceled, assuming that an appropriate plug-in is entered on the [account type](#)
- When the [effective date](#) of a frozen payment is changed
- If your [obligation](#) is governed by the **C1-FilingPeriodObligation** business object, there is a post processing plug-in that recalculates P&I if the override due date changes.
- When a [waiver](#) is activated or canceled
- An [overdue event](#) algorithm to calculate P&I is provided

The following events in the system request a forecast of P&I

- A determine schedule payments algorithm for a [pay plan recommendation rule](#) calls P&I to forecast the charges to the future scheduled payment dates.

## Contents

[C1-CALPI - Bring P&I Up to Date](#)  
[Adjustments and Updating P&I](#)

## C1-CALPI - Bring P&I Up to Date

Besides the events in the system that may cause a recalculation of P&I, the system also provides a background process [C1-CALPI](#) to periodically bring P&I up to date for all obligations.

The background process looks for all non-canceled non-closed obligations that reference a P&I control and invokes the appropriate P&I calculation algorithm.

## Adjustments and Updating P&I

It is important to note that P&I is NOT brought up to date any time an adjustment is frozen or canceled.

- Certain adjustments are created / canceled from the P&I calculation, for example the penalty and interest charges and waiver charges. For these types of adjustments, penalty and interest recalculation should not be invoked.
- For other adjustments where P&I should be brought up to date, there are many business scenarios where several adjustments are created for various charges. For example, when a tax form is processed, adjustments may be created for the tax assessment due and for the withholding credit and for refundable tax credits. The system should wait until all the adjustments are created before bringing P&I up to date, otherwise a lot of unnecessary calculations will occur.

Rather than building "update penalty and interest" logic into a plug-in for the adjustment, the expectation is that a business process that creates adjustments that affect P&I should include a step to invoke P&I when the adjustments are created. So following this practice, any adjustment that is created online in the [adjustment](#) page would not affect penalty and interest.

Also note that when an assessment adjustment is canceled, all related penalty and interest adjustments should be canceled as well. The P&I calculation does not cater for cleaning up penalty and interest for canceled assessments.

**Special service.** When cancelling an assessment adjustment, the business service **C1-CancelAssessmentAdj** should be used. This will clean up penalty and interest adjustments for the canceled assessment.

## Waivers

Waivers provide the ability to forgive or waive certain penalties, interest or fees. Sometimes waivers are referred to as abatements. You can waive a full amount, or partial amount (flat rates or percentages). Due to some legitimate hardship, waivers can be a one-time event, or in the case of natural disaster victims, waivers can be ongoing events.

### Contents

- [What is Waived?](#)
- [Waivers vs. Exemption](#)
- [Waivers and P&I Calculation](#)
- [Designing Your Waiver Options](#)

### What is Waived?

A waiver is created for a specific assessment and a specific debt category for that assessment. For example, interest for the obligation's original assessment may be waived or the failure to pay penalty for the amendment may be waived.

### Waivers vs. Exemption

A waiver is treated differently from an exemption in the base product algorithms. The assumption is that when an obligation is exempt from penalty and interest, calculations should not even be performed. Refer to [Eligibility](#) for more information.

For a waiver, the base product calculates the penalty or interest charge as normal and then creates appropriate waiver adjustments to offset the waived amounts. This allows an implementation to keep track of the amount of penalty and interest that has been waived.

### Waivers and P&I Calculation

In order for waivers to affect P&I calculation, appropriate algorithms must be plugged in. The following points highlight the algorithms that are required to support waiver functionality:

- Information about existing waivers should be retrieved at the [beginning of processing](#). A pre-processing algorithm is provided in the base product to populate information about the existing waivers for the obligation.
- Every P&I rule that calculates a charge that may be waived should include appropriate [rule processing](#) algorithms to adjust calculated charges in the internal P&I information data area based on existing waivers. The base product provides algorithms to support ongoing waivers and one-time waivers.
- [Post processing](#) algorithms should be plugged in to create or cancel waiver financial transactions when calling P&I in update mode.

**No excess waiver.** The base product algorithms assume that when a waived penalty or interest charge is later modified to cause the original amount to be less than the waiver, the waiver amount must also be adjusted accordingly. A waiver credit should never be in excess of the debt it's waiving.

## Designing Your Waiver Options

The base product provides admin and transaction business object pairs for one-time waivers (**C1-WaiverTypeOneTime** and **C1-OneTimeWaiver**) and for ongoing waivers (**C1-WaiverTypeOngoing** and **C1-OngoingWaiver**). Your implementation can add additional business rules to these BOs as required. If your implementation has waiver rules that aren't satisfied by one of the above business objects, you may create your own using the above as samples.

## Setting Up Penalty and Interest Options

---

The topics in this section describe how to set up the system to enable penalty and interest calculations.

### Contents

- Setting Up Account Types
- Setting Up Debt Categories
- Setting Up Debt Category Priorities
- Setting Up Adjustment Types
- Setting Up Adjustment Cancel Reasons
- Setting Up Rate Factors
- Setting Up Rate Quantity Indicators
- Setting Up Rate Schedules
- Setting Up P&I Control
- Setting Up P&I Rules
- Setting Up Obligation Types
- Setting Up Waiver Types

## Setting Up Account Types

In order for penalty and interest to be brought up to date when a payment is frozen or canceled, an appropriate

## Setting Up Debt Categories

Define [debt categories](#) for the following:

- One for each separate penalty or interest charge that is calculated by a system P&I rule.
- One for any other type of debt that is not calculated by P&I calculations, but is used in the calculation basis for one of the P&I rules.

- One for any other category of debts. For example, overpayments. When an obligation is overpaid, at some point the [overpayment](#) amount is reduced by one or more methods, such as minimum amount write down, carry forward to a future obligation, refund, etc. The adjustments created to reduce an overpayment are debits and therefore require a debt category. The suggestion is to create a special "Overpayment" debt category for these types of debts.

## Setting Up Debt Category Priorities

Define appropriate [debt category priorities](#) required by your implementation's business rules. Refer to [Debt Categories and their Priorities](#) for more information.

## Setting Up Adjustment Types

Adjustment types are needed for posting P&I charges.

- A separate adjustment type must be created for each debt category that is part of the P&I calculation.
- Each adjustment type created should use the **Penalty & Interest** adjustment type category.

Adjustment types are needed for posting waivers.

- A separate adjustment types must be created for each debt category whose P&I charges may get waived.
- Each adjustment type should refer to the **Waiver** adjustment type category.
- Each adjustment type should define the **C1-WAIVR** characteristic type as a valid template characteristic. This is used to reference the waiver that caused the adjustment to be created.

An adjustment type is needed for transferring cash accounting amounts from a "holding" general ledger account to the true "payable" account.

## Setting Up Adjustment Cancel Reasons

Define appropriate [adjustment cancel reasons](#) to use when cancelling P&I adjustments and based on recalculation and cancel reasons to use when cancelling waiver adjustments.

## Setting Up Rate Factors

Create appropriate [rate factors](#) that define the percentage to use when calculating your various P&I charges.

- If you are defining P&I rules using the base BO **C1-SimpleCalculationPIRule**, the P&I rule requires a rate factor to use when applying the charge.
- If you are defining P&I rules using the base BO **C1-MonthlyChargePIRule**, the P&I rule expects to use a Rate Schedule. You may choose to define rate factors for configuring the charges used in the specific rate components defined in this rate schedule.

## Setting Up Rate Quantity Indicators

Create appropriate [RQIs](#) required for your P&I rules. If you are defining P&I rules using the base BO **C1-MonthlyChargePIRule**, you need an RQI for the following information to pass into rate application:

- The calculation basis amount
- The not to exceed amount

## Setting Up Rate Schedules

Create appropriate [rate schedules](#) and their components to calculate the charges appropriately.

If you are defining P&I rules using the base BO **C1-MonthlyChargePIRule**, you need a rate schedule with rate components that calculate the charge by applying an appropriate percentage to the calculation basis passed in using the RQI defined above. If a Not to Exceed amount is applicable, there should also be a maximum rate component that adjusts the calculated charge based on the Not to Exceed amount.

## Setting Up P&I Control

A **P&I Control** is created to hold all the rules that work together to calculate automated / ongoing penalty and interest. The P&I control includes a list of **P&I Rules** that make up the individual calculations.

To set up P&I Control, select **Admin Menu, P&I Control**.

The topics in this section describe the base-package zones that appear on the P&I Control portal.

### Contents

- [P&I Control List](#)
- [P&I Control Actions](#)
- [P&I Control](#)
- [P&I Control Obligation Types](#)
- [P&I Rules](#)
- [P&I Control Log](#)

### P&I Control List

The P&I Control List zone lists every P&I Control. The following functions are available:

- Click a [broadcast](#) button to open other zones that contain more information about the adjacent P&I Control.
- Click the **Add** link in the zone's title bar to add a new P&I Control.

### P&I Control Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions and appropriate state transition buttons are available.



## P&I Control

The P&I Control zone contains display-only information about a P&I Control. This zone appears when a P&I Control has been broadcast from the P&I Control List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

## P&I Control Obligation Types

This zone lists obligation types currently linked to the broadcast P&I control.

## P&I Rules

This zone lists the P&I rules associated with the broadcast P&I control. Click on a P&I rule in the list to view its details in the [P&I rule portal](#).

If the P&I control is in **pending** status, Edit and Delete actions are available for each P&I rule. In addition you may add another P&I rule using the **Add** link in the zone's title bar.

## P&I Control Log

This is a standard [log zone](#).

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_PI\\_CTRL](#).

## Setting Up P&I Rules

This portal is provided to view and maintain details of a given P&I rule. This portal is not available from a menu. You navigate to this portal by drilling into a specific P&I rule from the [P&I control](#) portal.

The topics in this section describe the base-package zones that appear on the P&I Rule portal.

### Contents

- [P&I Rule Actions](#)
- [P&I Rule](#)

## P&I Rule Actions

This is a standard [actions zone](#). The Edit and Delete actions are available.

## P&I Rule

The P&I Rule zone contains display-only information about a P&I rule.

Please see the zone's help text for information about this zone's fields.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_PI\\_RULE](#).

## Setting Up Obligation Types

Each obligation type where penalty and interest charges are applicable must be configured appropriately:

- Define the default [debt category priority](#).
- It should include appropriate P&I controls that define the P&I rules that are in effect
- It should include appropriate P&I calculation algorithms. The following provides a list of the plug-in spots that must be required for P&I calculations. Unless noted, base algorithms are provided. If the functionality provided in the base algorithms does not satisfy your implementation's business rules, you may define your own:
  - P&I Calculation
  - Determine Balance Details
  - Retrieve FT Details
  - P&I Prepare Periodic Balance
  - P&I Pre-processing
  - P&I Post Processing - algorithms are provided for all algorithm types except for the [cash accounting](#) algorithm type. That one needs to be configured with the appropriate adjustment type.
- In addition to the required algorithms above, if your implementation's business rules include P&I Eligibility requirements, provide appropriate algorithms.

## Setting Up Waiver Types

To open the Waiver Type zone, select **Admin Menu, Waiver Type**.

### Contents

- [Waiver Type List](#)
- [Waiver Type](#)
- [Waiver Type Actions](#)
- [Waiver Type Log](#)

### Waiver Type List

The Waiver Type List zone lists every waiver type. The following functions are available:

- Click a [broadcast](#) button to open other zones that contain more information about the adjacent waiver type.
- Click the **Add** link in the zone's title bar to add a new waiver type.

### Waiver Type

The Waiver Type zone contains display-only information about a Waiver Type. This zone appears when a Waiver Type has been broadcast from the Waiver Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Waiver Type Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions and appropriate state transition buttons are available.

### Waiver Type Log

This is a standard [log zone](#).

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_WAIVER\\_TYPE](#).

# Defining Overpayment Processing Options

An overpayment occurs when the payment(s) exceeds the liability in an obligation, causing the obligation's balance to become a credit. Not every obligation that is in credit is considered to be an overpayment. This will typically depend on the tax type and the authority's rules.

## Contents

[The Big Picture of Overpayments](#)  
[Setting Up Overpayment Options](#)

## The Big Picture of Overpayments

---

For return-based taxes, such as individual income tax, a credit balance will be considered an overpayment only once a return is filed. Any payments or credits received in advance of a tax return are not considered overpayments until the tax form is processed, and an assessment is created. If a tax return is not filed, the obligation will be in a credit balance until the tax authority reviews it.

For billing-based taxes, different rules may apply to determine if an obligation is overpaid.

The following are common scenarios that would result in an overpayment:

- A majority of taxpayers receive refunds for their individual income tax filing because employer withholding tends to exceed the probable liability.
- Recalculation of tax, penalty, interest or fees that results in a reduced liability.

Not all tax types have overpayments. Assessments for sales tax and withholding tax are seldom overpaid because the taxpayer pays for an activity that has already happened. If an overpayment results, it is usually due to a math error on the form.

## Contents

[An Overpayment Process Can be Created Automatically or Manually](#)  
[An Overpayment is Typically Reviewed](#)  
[Not all Overpayments Result in a Refund](#)  
[Individual Taxpayer Overpayment Process](#)

## An Overpayment Process Can be Created Automatically or Manually

An overpayment process can be created in a variety of ways:

- As a result of posting a return. For example, the base Individual Income EZ Tax Form will create an overpayment process if the posting of the form results in an overpayment. See [Individual Income Tax Form](#) for how a form can be configured to create an overpayment process.

- An account monitor rule can be designed to create an overpayment process when the obligation's balance is a credit and a return has already been filed. The base-package provides the algorithm ***Initiate Overpayment Process*** ([C1-CC-INITOP](#)) that can be plugged in on the Collection Class Overdue Rules. For more information on how account monitor rules can be configured for overpayment, see [Overdue Rules Are Embodied In Algorithms](#).
- An overpayment process can be created manually by selecting **Main Menu, Financials, Overpayment Process**.

## An Overpayment is Typically Reviewed

When an overpayment is identified, it typically goes through a review process, which may include a combination of automated checks and user review and/or approval.

Examples of review activities include:

- Processing Rules - these can include checking for a valid mailing address, verifying bank account details.
- Compliance Rules - these include specific criteria defined by the tax authority for overpayments that require additional review.
- Minimum Balance - the overpayment amount is compared against a minimum threshold amount to determine whether the overpayment should be processed.
- Risk-based thresholds and overrides - if the overpayment amount is greater than a predefined maximum threshold, it may require special handling and additional approval.
- Suppression - an overpayment may be suspended if an account is under investigation for other debt or another reason.
- Separation of Duty - the tax officer approving the refund must be different from the tax officer who changed the account's bank information.

The review rules may differ with each tax authority and tax type. However, the end result is the same: to determine whether the overpayment can be processed further.

## Not all Overpayments Result in a Refund

Before an overpayment is refunded to the taxpayer, some or all of the overpaid amount may be offset against other existing debt, carried forward to a future period, and/or contributed to designated charitable funds or organizations.

### Contents

- [Calculating Interest](#)
- [Offset](#)
- [Carry Forward](#)
- [Contributions](#)
- [Refunds](#)

### Calculating Interest

Before any other action is taken, it might be required to calculate interest on the overpayment amount. The calculation rules will differ by tax authority and by tax types. The base-package is supplied with the following algorithms to calculate interest:

- Calculate interest using a rate-able adjustment ([C1-OP-CALINT](#))

- Calculate interest using a rate factor ([C1-OP-INTRF](#))

## Offset

When an overpayment is offset, the credit is used to extinguish other debts. The order of allocation of overpayment amount to debt depends on the tax authority's rules. Typically, the credit is applied in the following order:

- Within the same assessment
- Other assessments within the obligation
- Other obligations under the same account
- Other accounts for the taxpayer

In some cases, a tax authority may also allow offsetting against debt from other agencies. This type of debt is called external debt.

Where offset is possible, it is done before attempting to contribute, carry forward or refund. Offset is not allowed on future obligations, but a carry forward is done instead.

The base-package is supplied with an algorithm ([C1-OP-OFFST](#)) to offset credit to other obligations.

## Carry Forward

Taxpayers may opt to pay an amount of an overpayment to a future period. This is known as carry forward. This is common to individual income tax filing, where the taxpayer can indicate on the form a designated amount that will go to the next filing period. A taxpayer may also want to do a carry forward on an ongoing basis. In this case, the preferences might be kept somewhere in the account's information.

A tax authority's rules may designate certain tax types to only allow overpayments to be carried forward. Some common examples include excise or sales and use taxes. In these cases, the entire overpayment amount is carried forward.

The base-package is supplied with an algorithm ([C1-OP-CAFUFP](#)) to carry forward to a future period.

## Contributions

Most tax forms provide an option to contribute to a charitable fund/organization. The taxpayer indicates a specific amount to contribute to each selected fund/organization. The contribution is made only if an overpayment exists.

The base-package is supplied with an algorithm ([C1-OP-ACNAMT](#)) to create contributions.

## Refunds

Refunds result after attempts to offset, contribute and/or carry forward. The credit amount is given back to the taxpayer in the form of a paper check or a direct deposit. The taxpayer could either authorize a one-time direct deposit by putting bank account information on the tax return or indicate a recurring direct deposit for all refunds. In the case of a recurring direct deposit, the bank account information stored for the taxpayer on the [account](#) will be used.

The base-package is supplied with an algorithm ([C1-OP-CREREF](#)) to create a refund.

## Individual Taxpayer Overpayment Process

The base product includes the business object **C1-OvrpyProcAutoCredRef**, which is designed to cater for overpayment processes typical of individual taxpayers. The base business object **Individual Income EZ Tax Form** will create the overpayment process when the form indicates an overpayment amount.

### Contents

- [The Overpayment Process Lifecycle](#)
- [Overpayment Process Approval](#)
- [Enabling the System to Use Individual Income Overpayment Process](#)

### The Overpayment Process Lifecycle

The lifecycle of the overpayment process depends upon the configuration of the associated business object. The Individual Taxpayer Overpayment Process has a lifecycle typical of overpayments for individual taxpayers:

- The overpayment is initially validated against a number of business rules. The overpayment amount would typically be compared against a minimum threshold write-off amount.
- If issues are detected during validation, the overpayment will transition to an issues detected/error state. The user will be able to see a list of issues and correct them.
- The overpayment may require approval(s). The threshold amounts and approval roles will be defined on the overpayment type. A user can approve or reject the overpayment.
- Once all the required approvals have been obtained, certain actions will take place such as offset, carry forward, contributions and/or refund.

Refer to the **C1-OvrpyProcAutoCredRef** Business Object metadata for more details.

### Overpayment Process Approval

An overpayment process typically requires one or more levels of approval before any financial activity can take place.

### Contents

- [An Overpayment Process Typically Requires Approval](#)
- [To Do Entries Are Created To Notify Approvers](#)
- [Monitoring and Escalating Overpayment Approvals](#)
- [Rejecting Transitions the Overpayment Process to a Final State](#)

#### An Overpayment Process Typically Requires Approval

If the overpayment process type defines an approval hierarchy, users will need to approve the overpayment process in order for the process to continue. When the overpayment process is in the **Approval in Progress** state, two action buttons will appear: **Approve** and **Reject**.

If the user chooses to reject the overpayment process, they will be prompted for a reject reason.

### To Do Entries Are Created To Notify Approvers

When the overpayment process detects an approval is required, it notifies the first approver by creating a To Do entry. The To Do entry is created using the To Do type and To Do role defined on the overpayment process type. All users who belong to the approving To Do role can see the entry. When a user drills down on an overpayment approval To Do entry, the [overpayment process](#) portal is opened. This portal contains summary information about the overpayment process. This portal is also where the user approves or rejects the overpayment process.

When the user approves the overpayment process, the To Do entry is **Completed** and the overpayment process's log is updated. If there are no higher levels of approval required, the overpayment process will transition to the **Approved** state. If there are higher levels of approval required, a new To Do entry is created to the next To Do role in the approval hierarchy.

**To Do entries can create email.** A To Do entry can be configured to create an email message for every user in the To Do role to inform the user(s) of new overpayment processes requiring their attention. Refer to [To Do Entries May Be Routed Out Of The System](#) for the details.

### Monitoring and Escalating Overpayment Approvals

The base-package is supplied with an algorithm that your implementation can use to monitor overpayment approval requests that have been waiting too long for approval. This algorithm can complete the current To Do entry and create a new one for a different role when the timeout threshold defined on the algorithm's parameters is exceeded. If you've configured the system to send email for approval, this algorithm can also send x reminder emails (where x is defined on the algorithm's parameters) before the approval request is escalated to the new To Do role. Refer to [C1-OP-CHKOTO](#) for more information about this algorithm. If you plan to enable this functionality, plug in your configured algorithm on the **Approval In Progress** state on the **C1-OvrpyProcAutoCredRef** business object.

### Rejecting Transitions the Overpayment Process to a Final State

When an overpayment process is being approved, anyone with the right level of access can reject it by using the [overpayment approval](#) zone.

When an overpayment process is rejected, the following takes place:

- The user is prompted for a reject reason.
- The overpayment process's log is updated with the reject reason and the overpayment process is moved to the **Rejected** state.

### Enabling the System to Use Individual Income Overpayment Process

To enable this functionality the following configuration tasks are needed:

- Various adjustments are used for calculating interest, write off and offset for the overpayment process. For each of these actions, a suitable [adjustment type](#) is required.
  - If you wish to specify a minimum amount threshold for the overpayment process, you will need to define a non-calculated adjustment type for the minimum amount write-off.
  - If your implementation rules specify that interest should be calculated and the interest is calculated using a [rate](#), you will need to define the following:
    - A calculated adjustment type for offset-able interest, or a calculated adjustment type for non offset-able interest.
    - An associated rate schedule that includes a [RQ rule](#) for calculating number of days



- If your implementation rules specify that interest should be calculated and the interest is calculated using a [rate factor](#), you will need to define the following:
  - A non-calculated adjustment type for offset-able interest, or a non-calculated adjustment type for non offset-able interest.
  - An associated rate factor.
- If your implementation allows offset, then a non-calculated adjustment type needs to be defined for the offset.
- To allow refunds via a check, an [A/P adjustment type](#) needs to be defined.
- Make sure the adjustment types are defined on an associated [adjustment profile](#) that is linked to any obligation types that are going to be covered by the overpayment process.
- To allow refunds via direct debit a [distribution rule](#) needs to be defined. The distribution rule should be designed to pay a specific obligation.
- If approvals are required as part of the overpayment process, a to do type is needed. Note that the base-package is supplied with a to do type called [C1-OVAPP](#) that should be used as the basis for approval to do type.
- If you wish to have a to do created when the overpayment process is automatically cancelled, you will need a to do type. Note that the base-package is supplied with a to do type called [C1-OVPYX](#) that should be used as the basis for automatic cancellation to dos.
- If you wish to have a to do created when issues are detected during the validation of the overpayment process, you will need a to do type. Note that the base-package is supplied with a to do type called [C1-OVISS](#) that should be used as a basis.
- For each to do type that you wish to use, you will need a to do role.
- The system has been currently configured to calculate non offset-able interest using a rate factor. If you wish to change this you will need to inactivate the base-package algorithm by adding an appropriate inactivation option to the business object. You will then need to plug in a different algorithm based on the logic you wish to implement.
- Define an overpayment process type for the business object **C1-OvrpyProcTypeAutoCredRef**.

### Implementing Other Overpayment Process Types

The above sections describe how the base-package overpayment process for individual taxpayer works using the **C1-OvrpyProcAutoCredRef** and **C1-OvrpyProcTypeAutoCredRef** business objects. Your implementation can add additional business rules and change the overpayment process user interface as required. Alternatively, if your implementation needs a different overpayment process, you can create different business objects with their own business rules.

## Setting Up Overpayment Options

---

### Setting Up Overpayment Process Types

An **Overpayment Process Type** defines the configuration information that is common to overpayment processes of a given type. The type of information captured on the overpayment process type is governed by the overpayment process type's business object.

To set up an Overpayment Process Type, select **Admin Menu, Overpayment Process Type**.

The topics in this section describe the base-package zones that appear on the Overpayment Process Type portal.

### Contents

- [Overpayment Process Type List](#)
- [Overpayment Process Type Actions](#)
- [Overpayment Process Type](#)
- [Overpayment Process Type Log](#)

## Overpayment Process Type List

The following functions are available:

- Click a [broadcast](#) button to open other zones that contain more information about the adjacent overpayment process type.
- Click the **Add** link in the zone's title bar to add a new overpayment process type.

## Overpayment Process Type Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions and appropriate state transition buttons are available.

## Overpayment Process Type

The Overpayment Process Type zone contains display-only information about an Overpayment Process Type. This zone appears when an Overpayment Process Type has been broadcast from the Overpayment Process Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_OP\\_PROC\\_TYPE](#).

## Overpayment Process Type Log

This is a standard [log zone](#).

# Defining Obligation Types

Every obligation must reference an obligation type. The obligation type defines the responsibilities between the taxpayer and the tax authority. The obligation type dictates how returns, bills and payments are handled as well as how overdue debt is collected and how tax collections will be booked in your general ledger.

## Contents

- [Designing Obligation Types](#)
- [Setting Up Obligation Types](#)
- [Setting Up Terms and Conditions](#)
- [Setting Up Obligation Type Start Options](#)
- [Obligation Type Start Options Merge](#)

## Designing Obligation Types

---

The topics in this section provide guidelines describing how to design obligation types. When designing obligation types you will want to consider what the obligation will be used for. Generally speaking, obligation types can be separated into two categories.

- The first type of obligation type relates to the specific tax types of a tax authority. Examples of these include corporate tax and property tax. The specific tax type dictates how bills are handled and what collections activities are available to the tax authority.
- The second type of obligation relates to specific events or transactions that require certain follow-up actions from the tax authority or taxpayer (or both). Examples of these include overpayment obligations and payment plan obligation. With overpayments, the tax authority has the responsibility to properly handle the excess credit a taxpayer has created with a payment. Payment plans are the responsibility of the taxpayer in that they are required to send in a series of payments on a predefined schedule. Each obligation must be monitored to ensure proper compliance.

## Contents

- [Filing Period Obligations](#)
- [Property Tax Obligations](#)
- [Billable Charge Obligations](#)
- [Overpayment Obligations](#)
- [Write Off Obligations](#)
- [Over/Under Cash Drawer Obligations](#)
- [Payment Upload Error Obligations](#)
- [Pay Plan Obligations](#)

## Filing Period Obligations

Filing period obligations are one of the tax type specific obligation types. Examples of these tax types include individual income, sales and use, corporate, withholding, and fuel taxes. These are examples of return based taxes in which the taxpayer has the responsibility to file a return for each filing period obligation. There are usually statutory considerations when configuring these obligation types. You will want to consider the following when designing your obligation for these types of taxes.

- The obligation type must reference a defined [tax type](#). The specific tax type will dictate the rules for that obligation. It should indicate that a Tax Role is required and should include the valid filing calendars for the tax type.
- The obligation type should be configured to indicate that a filing period is required.
- Distribution code and general ledger division for the obligation directs how the money is accounted. Different taxes may be moved to different accounts. While an individual income tax may go to the general treasury fund, a fuel tax may go the highway construction fund. The distribution code will reference an algorithm that directs how the money will be accounted for.
- Adjustment profile must be defined for the obligation. Adjustment profiles contain the individual listing of adjustment types that are available for that obligation type.
- An appropriate P&I control and P&I algorithms should be defined for the obligation type. Refer to [P&I Rules for a P&I Control Define the Calculation](#) for more information.
- Define the default debt category priority for credits linked to obligations of this type. Refer to [Debt Categories and their Priorities](#) for more information.

Each tax type will have its own unique attributes that will change how the obligation is set up. The more details you add to your obligation type, the more robust your processing of that obligation will be. Before configuring for this obligation type you should thoroughly review the statutory considerations and make sure they are all addressed when setting up your obligation type.

## Property Tax Obligations

Property tax obligations are another type of tax specific obligation type. Property tax obligations are based off the assessed value of a taxpayer's asset. Typically they are items such as real estate or automobiles but can include boats, RV's, and other high value items. They differ from filing period obligations in that it is the tax authority's responsibility to assess the value and calculate the amount of tax owed. It is then the taxpayer's responsibility to either pay or appeal the assessed value. The considerations are very similar to filing period obligations.

- The obligation type must reference a defined tax type. The specific tax type will dictate the rules for that obligation.
- Payment plans are often utilized for property tax obligations. Due to their infrequent assessment and billing (once or twice a year) the amount is usually large enough where a significant number of taxpayers would have difficulty paying. If payment plans are utilized then make sure to check that option under the billing tab.
- Distribution code and general ledger division for the obligation directs how the money is accounted. Different taxes may be moved to different accounts. While an individual income tax may go to the general treasury fund, a fuel tax may go the highway construction fund. The distribution code will reference an algorithm that directs how the money will be accounted for.
- Rates may be utilized more often for property taxes. Property taxes tend to be a locally administered tax. The breakdown of how the tax is applied to local services can be done using the rate schedule option under the rate tab.
- Adjustment profile must be defined for the obligation. Adjustment profiles contain the individual listing of adjustment types that are available for that obligation type.

## Billable Charge Obligations

You create a billable charge whenever a taxpayer should be charged for a service that occurs outside the normal course of business. For example, if a taxpayer requires a review of their property assessment, you may charge them an administration fee. You can also use billable charges to “pass through” other bill ready charges generated outside the system, by another application, or by a 3<sup>rd</sup> party supplier.

A billable charge must reference an obligation. This obligation behaves just like any other obligation:

- **Bill segments are created for the obligation.** Whenever billing is performed for an account with billable charge obligations, the system creates a bill segment for each obligation with unbilled charges. If multiple unbilled charges exist for a given obligation, only one bill segment will be created and it will contain details about all of the billable charges.
- **Payments are distributed to the obligation.** Payments made by an account are distributed to its billable charge obligations just like any other obligation.
- **Overdue debt is monitored.** The credit and collections process monitors billable charge obligations for overdue debt and responds accordingly when overdue debt is detected.

Therefore, you must set up at least one obligation type to hold your billable charge debt. You may have multiple charges based on billing frequencies, A/R booking, debt monitoring, etc. It's really up to you.

The easiest way to determine how many billable charge obligation types you'll need is to define every conceivable billable charge (which you should have done when you designed your billable charge templates). Then ask yourself if they have the same billing and payment behavior, if so, you'll have one obligation type. If not, you'll need one obligation type for each combination.

## Overpayment Obligations

When a taxpayer pays more than they owe, you must decide what to do with the excess money. The following points describe two possibilities:

- You could create a new obligation to hold the excess (let's call it an overpayment obligation). The credit would need to be transferred from this obligation to the tax obligations at an appropriate time.
- You could direct the excess payment on one of the existing tax obligations.

You control which method is used by plugging in the appropriate **Overpayment Distribution** algorithm on each [account type](#) (i.e., you can choose a different method for different account types). If you choose to hold overpayments on a separate obligation, then you must set up an obligation type.

The following points highlight interesting information about overpayment obligation types:

- It should have an accounts payable distribution code. This is because when a payment segment is created for this type of obligations, the system must credit a liability (an overpayment is a liability).
- It's important to indicate that the overpayment obligation is a one-time obligation. Why? Because this means that the system will automatically close the obligation when its balance falls to zero (i.e., when appropriate business processes transfer all of the overpayment to tax obligations).

- A bill segment type is not needed because the system never creates bill segments for such obligations (they exist only to hold excess credits).
- P&I controls and the P&I related algorithms are not needed.
- You may also want to turn on the alert message
- You must reference this overpayment obligation type as the parameter value on your overpayment algorithm (this algorithm is plugged in on the desired account types). Refer to [Overpayment Algorithm](#) for more information about this algorithm.
- You must design appropriate business procedures to use this overpayment when additional debt is incurred on tax obligations.

## Write Off Obligations

Some agencies may choose to transfer written off debt from the “normal” obligation onto one or more write-off obligations. When the debt is transferred to a write-off obligation, the distribution code on the “normal” obligation is credited (typically an A/R GL account), and the distribution code on the write-off obligation is debited.

You will almost always need a write-off obligation whose distribution code is the write-off expense. However, if you don't practice cash accounting, you may have uncollectible debt for liabilities (and you don't owe the liability if you don't get paid). In this case you'll need another obligation type for the liabilities.

The following points highlight interesting information about write-off obligation types:

- The distribution code is either an expense or a liability account. This is because when debt is transferred to these types of obligations, the system must debit either an expense account (i.e., write-off expense) or a liability account. It's important to note that in The Ramifications of Write Offs in the General Ledger we explain how this liability account may be overwritten with the liability account that was originally booked.
- They do not need a bill segment type because the system never creates bill segments for such obligations (they exist only to hold uncollectable debt)

**Note.** The adjustment type used to set the offending obligation's current balance equal to its payoff balance is defined on each write-offable obligation type. The adjustment type used to transfer the delinquent debt to the write-off obligation is defined on the write-off obligation type.

**An Alternative.** If you have a limited number of liability accounts, you may decide to have a separate write-off obligation for each liability account. Doing this would proliferate the number of obligations created at write-off time. However, it would simplify the remittance of payment to the third party if the reversed liability is ever paid.

## Over/Under Cash Drawer Obligations

In order to balance a tender control that is out-of-balance, your organization must set up an account with an obligation whose obligation type references the over/under expense account. You will probably only have one obligation that references this obligation type, but you still must have it if you remit funds via a cash drawer.

For more information about over/under processing, refer to [How To Get An Unbalanced Tender Control In Balance \(Fixing Over/Under\)](#).

The following points highlight interesting information about this obligation type:

- It has an expense distribution code. This is because when a payment segment is applied to this type of obligation, the system must debit an expense account for under amounts (and credit it for over amounts).
- It doesn't need a bill segment type because the system never creates bill segments for such obligations (it only has over/under payment segments linked to it).

## Payment Upload Error Obligations

If the payment upload process detects an invalid account on a payment upload record, it will create a payment for the suspense obligation defined on the upload process' tender source (see [Setting Up Tender Sources](#)). You should create a special obligation type for this obligation.

For more information about the payment upload process, refer to [Phase 3 – Create Payment Events, Tenders, Payments and Payment Segments](#).

The following points highlight interesting information about this obligation type:

- It has an expense distribution code. This code should probably be a suspense account. All payment segments that are created for this obligation will eventually be transferred to a "real" obligation and therefore this GL account's balance should be zero when no payments are in suspense.
- It doesn't need a bill segment type because the system never creates bill segments for such obligations (it only has invalid account payment segments linked to it).

## Pay Plan Obligations

If you allow your customers to pay overdue debt using a payment plan, you need to set up obligation types for pay plan obligations.

For more information about pay plans, refer to [Defining Pay Plan Options](#).

## Setting Up Obligation Types

---

In this section, we explain how to create/maintain your obligation types.

### Contents

- [Obligation Type - Main Information](#)
- [Obligation Type - Detail](#)
- [Obligation Type - Billing](#)
- [Obligation Type - Rate](#)
- [Obligation Type - Adjustment Profiles](#)

[Obligation Type - Characteristics](#)  
[Obligation Type - Algorithms](#)  
[Obligation Type - Pay Plan Recommendation Rule](#)  
[Obligation Type - P&I Control](#)  
[Obligation Type - BC Template](#)  
[Obligation Type - BC Upload Overrides](#)

## Obligation Type - Main Information

Open **Admin Menu**, **Obligation Type** and navigate to the **Main** tab to define core information about your obligation types.

### Description of Page

Enter a unique combination of **Division** and **Obligation Type** for every obligation type.

Enter a **Description** for the obligation type.

**Tax Type** defines the type of service associated with the obligation type. If the obligation type has rates, only rates belonging to this tax type may be linked to the obligation type.

For more information about tax types, refer to [Setting Up Tax Types](#).

Use an **Obligation Business Object** to define a [BO](#) that may govern additional rules related to obligations of this type.

Indicate if the **Filing Period Applicability** for obligations of this type is **Required** or **Not Allowed**.

Select the **Distribution Code** and **GL division** that defines the receivable account for receivable-oriented obligations. For non-receivable oriented obligations, this distribution code depends on the obligation type's function. Refer to [Designing Obligation Types](#) for some examples.

Select the **Revenue Class** associated with the obligation type (and its obligations). The revenue class may affect the revenue account(s) generated by the obligation's rate.

Refer to [Rate Component – GL Distribution](#) for more information about revenue class.

Select the **Payment Segment Type** that defines how payment segments linked to obligations of this type affect:

- The obligation's payoff and current balances

For more information about payment segment types, refer to [Setting Up Payment Segment Types](#).

When a tender is canceled, a cancellation reason must be supplied. If the cancellation reason indicates a NSF (non sufficient funds) charge should be levied, the system invokes the Levy an NSF Charge algorithm specified on the tender's account's [account type](#). Because adjustments must be linked to an obligation, the algorithm must determine the appropriate obligation to use to levy the adjustment based on business rules. The charge is levied using the **NSF Adjustment Type** of the appropriate obligation's obligation type.



**Warning!** You must specify adjustment type profiles on the obligation type (on the Adjustment Type window) before adjustment types will appear in the above drop downs.

For more information about adjustment types, refer to [Setting Up Adjustment Types](#).

Define the **Payment Grace Days** applicable for obligations of this type. Refer to [Due Dates for Filing Period Based Obligations](#) for more information about due dates.

Select the **Payment Priority**. This field is available for use by the algorithms that distribute partial payments amongst an account's obligations. Higher priority obligations will have their debt relieved before lower priorities. Refer to [Distribution Based on Payment Priority](#) and [Delinquent Payment Distribution Algorithm](#) for information about payment distribution algorithms that use this field.

**Note.** The values for this field are customizable using the Lookup table. This field name is PAY\_PRIORITY\_FLG.

For more information about distribution priority, refer to [Distributing A Payment Amongst An Account's obligations](#).

Select the **Delinquent Payment Priority**. This field is available for use by the algorithms that distribute partial payments amongst an account's obligations. Higher priority obligations will have their debt relieved before lower priorities. Refer to [Delinquent Payment Distribution Algorithm](#) for information about a payment distribution algorithm that uses this field.

**Note.** The values for this field are customizable using the Lookup table. This field name is DEL\_PRIORITY\_FLG.

Define the default **Debt Category Priority** for credit financial transactions linked to obligations of this type. Refer to [Debt Categories and their Priorities](#) for more information.

Turn on **Do Not Overpay** if the system is not allowed to distribute an overpayment to this type of obligation (i.e., the obligation is not allowed to have a system-created credit balance). This field is available for use by algorithms that distribute overpayments. Refer to [Overpayments Held On Highest Priority Obligation](#) for information about an overpayment algorithm that uses this field.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_SA\\_TYPE](#).

## Obligation Type - Detail

Open **Admin Menu, Obligation Type** and navigate to the **Detail** tab to define additional details about a given obligation type.

#### Description of Page

Turn on **Display As Alert** if Control Central should display an alert if an account has an obligation of this type that isn't **Closed** or **Canceled**. If this switch is on, also enter the **Alert Information** to appear on Control Central. We recommend only using this feature on unusual obligation types (e.g., write-offs) so that a CSR is not presented with an alert for every obligation type.

If this obligation type is used for any of the special roles defined in the drop down of **Special Role Flag**, indicate which one. Valid values are: **Billable Charge**, **Pay Plan**, **Write Off**. This information is used on windows with functionality that can only be used by obligations used for specific roles. For example, the Billable Charge window group can only reference **Billable Charge** obligations.

If the Special Role is **Write Off**, you must also define the following adjustment types:

- Use **Adjustment Type (Xfer)** to specify the type of adjustment used to transfer funds from the uncollectable obligations to the write off obligation.

**Warning!** You must specify adjustment type profiles on the obligation type (on the Adjustment Type window) before adjustment types will appear in the above drop downs.

Turn on **One Time Charge** if this obligation type is used for one-time invoices. When a one-time invoice obligation is created, the system sets the stop date of the obligation to be equal to the start date.

### Where Used

The alert information is used by Control Central to alert a CSR when unusual obligations exist for an account. Refer to [Control Central – Main](#) for more information.

Only obligation types designated as being **Billable Charge** may have billable charges linked to them. Refer to [Maintaining Billable Charges](#) for more information.

Only obligation types designated as **Pay Plan** may be used to set up pay plans. Refer to [Pay Plans](#) for more information.

Only obligation types designated as being **Write Off** may be specified as the write off obligation type on distribution codes. Refer to [Setting Up Distribution Codes](#) for more information.

## Obligation Type - Billing

Open **Admin Menu**, **Obligation Type** and navigate to the **Billing** tab to define how the system manages bill segments for obligations of a given obligation type.

### Description of Page

Turn on **Eligible for Billing** if the system should create bill segments for obligations of this type.

Define the minimum number of days a bill segment (other than the final segment) must span using **Minimum Days for Billing**. This is useful to prevent initial bill segments that span only a few days.

For more information about minimum days, refer to [Preventing Short Bill Segments](#).

Select the **Bill Segment Type** that controls both how bill segments for this obligation type will be created and how the related financial transaction affects the general ledger and the taxpayer's debt.

For more information about bill segment types, refer to [Setting Up Bill Segment Types](#).

Define the default **Debt Category** to assign to bill segment type financial transactions for obligations of this type. Refer to [Debt Categories and their Priorities](#) for more information.

**Required for base algorithms.** The base P&I calculation algorithm and the base [Determine Detailed Balance](#) algorithm require that every debit financial transaction reference a debt category.

Use **Default Description on Bill** to define the verbiage that should print on the taxpayer's bill.

**Rates overwrite this description.** The Default Description on Bill is not applicable for obligations whose charges are calculated using a rate. Why? Because the description that appears on the bill segment is defined on the rate schedule's rate version.

**Billable charges overwrite this description.** The Default Description on Bill is not applicable for obligations whose charges are calculated using a billable charge. Why? Because the description that appears on the bill segment is defined on the billable charge.

Use the **Billing Processing Sequence** if you need to control the order in which obligations linked to this obligation type are processed by billing processes.

Use **Bill Print Priority** to define the order in which the obligation type's bill segments should appear on bills (relative to the other obligation types that appear on a bill).

**Note.** The values for this field are customizable using the Lookup table. This field name is BILL\_PRT\_PRIO\_FLG.

Use **Max Bill Threshold** if you want the system to generate a bill error when a bill segment is produced in batch that exceeds a given value. These bill errors will appear on the standard billing queries and To Do lists. If, after reviewing the high value bill segment, an operator truly intends to send the bill out, they should regenerate the bill. Refer to [How To Correct A Bill Segment That's In Error](#) for more information.

**Warning!** The value entered in this field will DEFAULT onto obligations of this type when they are first created. An operator may change the default value on an obligation in case a specific taxpayer has unusually high bills that continually error out. It's important to be aware that if you change the value of High Bill Amount on an obligation type and there already exist obligations of this type, the existing obligations will contain the original value (the new value on the obligation type will not be propagated on the existing obligations).

Turn on **Characteristic Location Required** if a characteristic location must be linked to the obligation when the obligation is activated. The characteristic location is used to define the taxing authorities associated with the obligation's bill segments. It is also used to identify where the obligation's service is located on various windows.

For more information about how characteristic location is used, refer [An Illustration Of A Rate Factor And Its Characteristics](#).

Use the **Initial Start Date Option** to control how billing should calculate the calculation period for the very first bill for obligations of this type. This field is important if your obligation has rate components that include daily charges. Set the field to **Include First Day** if the obligation's start date should be included in the daily charges. Set the value to **Add 1 Day Always** if the obligation's start date should never be included in the daily charges. Set the value to **Add 1 Day for Back-to-back** if the obligation's start date should only be included in the daily charges if no previous taxpayer was responsible for the charge (for example for property tax). This field is not applicable for obligation types with a special role of **Billable Charge**.

Obligations may have the end date of their bill segments defined on a user-maintained bill period schedule. This option is used when bill segments must fall on strict calendar boundaries (e.g., quarterly bills that end on the last day of the quarter). If this obligation type should be billed like this, select **Use Bill Period** in the **Use Calendar Billing** field. When this option is used, you must define the **Bill Period** whose schedule defines the bill segment end dates.

For more information about bill period schedules, refer to [Defining Bill Cycles and Bill Periods](#). For more information about other bill end date methods, refer to [Ways To Control The End Date Of A Bill Segment](#).

Instead of the **Use Bill Period** method, obligations may have their bill segment end date based on a specified date. If this obligation type should be billed like this, select **Anniversary Future Billing** or **Anniversary Past Billing** in the **Use Calendar Billing** field. When either option is used, you must define the **Anniversary Bill Frequency**. This frequency defines the amount of time between bill segments.

For more information about anniversary billing, refer to [Using The Anniversary Method](#). For more information about other bill end date methods, refer to [Ways To Control The End Date Of A Bill Segment](#).

**Total Bill Amount** indicates whether obligations of this type can use the total amount to bill field on the obligation page. Valid values are **Not Allowed** and **Required**.

If **Required** is selected, you must enter the **Total Amount To Bill Label**. The **Total Amount To Bill Label** defines the label that prefixes the total bill amount on the obligation page for obligations of this obligation type.

**Recurring Charge** indicates whether obligations of this type can use the recurring charge field on the obligation window. Valid values are **Not Allowed**, **Optional** and **Required**. If either **Optional** or **Required** are used, you must enter:

- **Recurring Charge Amount Label.** This defines the label that prefixes the recurring charge amount on the obligation window for obligations of this obligation type.
- **Recurring Charge Frequency.** This defines the following:
  - Specifies the frequency at which the Recurring Charge Amount specified on obligations of the obligation type is to be billed.
  - Serves as the basis for proration of the Recurring Charge Amount.
  - Specifies the frequency at which obligations of the obligation type without a rate will be billed.

Set the **Eligible for Pay Plan** flag to **Eligible** if you want obligations of this type to be eligible to be covered by a pay plan.

## Obligation Type - Rate

Open **Admin Menu, Obligation Type** and navigate to the **Rate** tab to define the rates that may be referenced on obligations of a given type.

### Description of Page

Turn on **Rate Required** if the bill segment creation algorithm for the obligation type expects a rate schedule to be referenced on obligations of this type.

For more information, refer to [Rates](#).

Define the date the system uses when selecting an effective-dated rate (from the obligation's rate history) using **Rate Selection Date**. Selecting **Bill Start Date** will cause the system to use the rate effective on the first day of the bill segment's calculation period. Selecting **Bill End Date** will cause the system to use the rate effective on the last day of the bill period. Selecting **Accounting Date** will cause the system to use the rate effective on the accounting date associated with the bill.

The information in the **Rate Schedules** collection defines the rates that may be referenced on obligations of this type. The following fields are required for each obligation type:

<b>Rate Schedule</b>	Specify the rate schedule; its description is displayed adjacent.
<b>Use Rate As Default</b>	Turn on this switch for the rate to be defaulted on new obligations.

### Where Used

This information is used to default and validate the rate specified on an obligation. Refer to [Obligation – Rate Info](#) for more information.

## Obligation Type - Adjustment Profiles

Open **Admin Menu, Obligation Type** and navigate to the **Adj Profile** tab to define the adjustment profiles that define adjustment types that may be referenced on obligations of a given type.

### Description of Page

Define the **Adjustment Type Profiles** that, in turn, define adjustment types that may be referenced on obligations of a given type.

For more information about adjustment type profiles, refer to [Setting Up Adjustment Type Profiles](#).

**Where Used**

This information is used to validate the adjustments linked to the obligation. Refer to [Adjustments – Main Information](#) for more information.

## Obligation Type - Characteristics

To define characteristics common to all obligations of a given type, open **Admin Menu**, **Obligation Type** and navigate to the **Characteristics** tab.

**Description of Page**

Use the characteristics collection to define a **Characteristic Type** and **Characteristic Value** common to all obligations of this type.

**Note.** You can only choose characteristic types defined as permissible on an obligation type record. Refer to [Setting Up Characteristic Types & Their Values](#) for more information.

## Obligation Type - Algorithms

Open **Admin Menu**, **Obligation Type** and navigate to the **Algorithm** tab to define the algorithms that should be executed for obligations of a given type.

**Description of Page**

The grid contains **Algorithms** that control important functions in the system. You must define the following for each algorithm:

- Specify the **System Event** with which the algorithm is associated (descriptions of all possible events are provided below).
- Specify the **Sequence** and **Algorithm** for each system event. You can set the **Sequence** to 10 unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

**Warning!** These algorithms are typically significant processes. The absence of an algorithm may prevent the system from operating correctly.

The following table describes each **System Event** for which you can define algorithms.

System Event	Optional / Required	Description
<i>Bill Completion</i>	Optional	<p>These algorithms are executed whenever a bill is completed for an account that contains a non-canceled obligation of this type.</p> <p><b>Note.</b> Algorithms of this type are called for all non-<i>Canceled</i> obligations, regardless of whether or not they are billed. If your algorithms should only be processed under certain conditions (for example, only process this algorithm for <i>Active</i> obligations), then it is the responsibility of the algorithm to check the conditions before continuing.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Break Pay</i>	Optional	These algorithms are executed when a <a href="#">pay plan</a> is manually stopped via the <a href="#">pay</a>

<i>Plan Obligation</i>		<p><a href="#">plan maintenance page</a>.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Determine Detailed Balance</i>	Optional	<p>This plug-in spot is responsible for breaking down an obligation's balance into balances by separate assessments and debt categories (for example: tax, penalty, interest, fees, etc). It is used in the base <i>P&amp;I Calculation</i> plug-in and several other services that require the balance of an obligation to be broken down by assessment and debt category.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.</p>
<i>FT Freeze</i>	Optional	<p>These algorithms are executed whenever a financial transaction is frozen that is linked to an obligation of this type.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Obligation Activation</i>	Optional	<p>These algorithms are executed when an obligation status changes from <i>Pending Start</i> to <i>Active</i>. It performs any additional activities that are necessary to activate an obligation.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Obligation Cancel</i>	Optional	<p>These algorithms are executed when an obligation status changes to <i>Canceled</i>. It performs any additional activities that are necessary when an obligation is canceled.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Obligation Creation</i>	Optional	<p>These algorithms are executed when an obligation is created.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Obligation Information</i>	Optional	<p>We use the term "obligation information" to describe the basic information that appears throughout the system to describe an obligation. The data that appears in "obligation information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the "obligation information" algorithm on installation options or the system default "obligation information" if no such algorithm is defined on installation options.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Obligation Stop</i>	Optional	<p>These algorithms are executed whenever an obligation's status changes from <i>pending stop</i> to <i>stopped</i>.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>Obligation Stop Initiation</i>	Optional	<p>These algorithms are executed whenever an obligation's status becomes <i>pending stop</i>.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>
<i>P&amp;I Calculation</i>	Optional	<p>This plug-in spot is responsible for <a href="#">calculating penalty and interest</a> for the input obligation.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.</p>
<i>P&amp;I Eligibility</i>	Optional	<p>This plug-in spot is invoked by the base Calculate Penalty and Interest plug-in. It receives an obligation id as input and returns an indication of whether the obligation is eligible or ineligible for P&amp;I.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event and for more</p>



		information about the behavior of this plug-in spot.
<i>P&amp;I Post Processing</i>	Optional	This plug-in spot is invoked by the base P&I Calculation plug-in after all the penalty and interest calculations are finished. Algorithms plugged into this plug-in spot are responsible for additional logic performed at the end of the P&I process. Click <a href="#">here</a> to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.
<i>P&amp;I Pre-processing</i>	Optional	This plug-in spot is invoked by the base P&I Calculation plug-in and is used to populate information needed during P&I calculations. Click <a href="#">here</a> to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.
<i>P&amp;I Prepare Periodic Balance</i>	Optional	This plug-in spot is called from the Calculate Penalty and Interest plug-in spot for each date range being calculated. It is responsible for passing the appropriate FTs to the <i>Determine Detailed Balance</i> plug-in for the current date range. Click <a href="#">here</a> to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.
<i>Payment Freeze</i>	Optional	These algorithms are executed whenever a payment is frozen. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Process Pay Plan Scheduled Payment</i>	Optional	These algorithms are executed by the Pay Plan Scheduled Payment Processing background process whenever a scheduled payment is due. If the pay plan obligation is unmonitored, this algorithm is not called. This algorithm should be specified on pay plan obligation types to create the necessary adjustments for the pay plan obligation. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Retrieve FT List</i>	Optional	This plug-in spot is responsible for retrieving the existing FTs for an obligation. It obtains all the data for each FT that is needed for <i>P&amp;I Calculation</i> logic and the <i>Determine Detailed Balance</i> logic. Click <a href="#">here</a> to see the algorithm types available for this system event and for more information about the behavior of this plug-in spot.

## Obligation Type - Pay Plan Recommendation Rule

Open **Admin Menu**, **Obligation Type** and navigate to the **Pay Plan Rec'n Rule** tab to define the recommendation rules that are valid for pay plan obligations of this type.

### Description of Page

If the obligation type's special role is *Pay Plan*, you may define the **Recommendation Rules** that are valid on pay plan obligations of this type. Check the **Use As Default** box to indicate the default recommendation rule for obligations of this type.

For more information about pay plans, refer to [Defining Pay Plan Options](#).

### Where Used

The pay plan recommendation rules are used to recommend the payment amount and payment schedule for pay plan obligations. Refer to [Maintaining Pay Plans](#) for more information.



## Obligation Type - P&I Control

Open **Admin Menu, Obligation Type** and navigate to the **P&I Control** tab to define the penalty and interest control record that governs the [penalty and interest calculation](#) rules for obligations of this type.

### Description of Page

The P&I Control and its P&I Rules define the configuration governing the tax authority's penalty and interest calculations. As these rules change over time, a new P&I Control record with the new applicable P&I Rules is created and linked to the applicable obligation types for the effective date of the change.

The following fields display:

- **Effective Date** is the date the P&I Control rate becomes effective.
- **P&I Control** defines the P&I rules used to calculate the individual penalty and interest charges.

**P&I configuration.** Refer to [Setting Up Penalty and Interest Options](#) for more detail about the configuration required for calculating penalty and interest.

### Where Used

This information is used by the P&I Calculation plug-in to apply specific rules.

## Obligation Type - BC Template

Open **Admin Menu, Obligation Type** and navigate to the **BC Template** tab to define the billable charge templates that can be used on obligations of a given type.

**Only billable charges have billable charge templates.** Only obligations that are defined as Billable Charges (in the Special Role on the Details window) may use the grid on this window.

### Description of Page

The information in the **Billable Charge Template** collection defines the obligation type's permissible billable charge templates. A billable charge template contains the default bill lines, amounts and distribution codes used to levy a one-off charge. The following fields are required for each template:

#### Billable Charge Template

Specify the billable charge template. Its description is displayed adjacent.

#### Use As Default

Turn on this switch for the template to be defaulted on new billable charges linked to obligations of this type (if any).

### Where Used

This information is used to limit the billable charge templates that can be used for a given obligation type.

## Obligation Type - BC Upload Overrides

The [BCU2 – Create Billable Charge](#) background process is responsible for creating billable charges for each billable charge upload staging record interfaced into the system. This process will override the values of the various switches referenced on a bill charge upload staging line if the respective obligation's obligation type has an override value for the bill charge upload staging line's billable charge line type.

**This information is optional.** If you don't need to override the values of a [Billable Charge Line Type](#) you don't need to set up this information.

Open **Admin Menu, Obligation Type** and navigate to the **BC Upload Override** tab to define override values for a given Obligation Type / Billable Charge Upload Staging Line Type.

### Description of Page

Use the **Billable Charge Overrides** collection to define values to be overridden on billable charge lines uploaded from an external system (refer to the description above for the details). The following switches may be overridden for a given **Obligation Type** and **Billable Charge Line Type**.

- Use the **Show on Bill** switch to define the value to be defaulted into the Show on Bill indicator on billable charge upload lines that reference this line type.
- Use the **Appear in Summary** switch to define the value to be defaulted into the App in Summary indicator on billable charge upload lines that reference this line type.
- Use **Memo Only, No GL** switch to define the value to be defaulted into the Memo Only, No GL indicator on billable charge upload lines that reference this line type.
- Use **Distribution Code** to define the value to be defaulted into the Distribution Code on billable charge upload lines that reference this line type.

## Setting Up Terms and Conditions

---

Your obligation type start options can reference terms and conditions (T&C's) that should be defaulted onto new obligations. Each T&C is identified with a terms and condition code. To define a terms and conditions code, open **Admin Menu, Terms and Conditions**.

### Description of Page

Enter a unique **Terms and Condition** code and **Description**. Use **Text** to describe the exact terms.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_TC](#).

## Setting Up Obligation Type Start Options

Obligation type start options save users time and prevent data entry errors because they default many values on an obligation (e.g., the rate schedule, recurring charge amount, contract riders, contract terms, characteristics, terms and conditions, etc. can all be defaulted onto an obligation from a start option).

An obligation type may have zero or more start options.

- An obligation type without start options is usually one that has a very limited number of options. For example, if an obligation type has a single valid rate and no taxpayer-specific contract values, you don't need to setup a start option (the obligation's default rate can default based on the information defined when you setup the obligation type).
- An obligation type with multiple start options is one where many different permutations are possible. For example, an obligation type that can have multiple rates and each rate can have multiple riders is a good candidate for start options (where each start option will default, for example, a specific rate and set of contract riders).

A start option's default values may change over time (i.e., the information on a start option is effective-dated).

Start options can cause a great deal of information to be populated on an obligation. There are several ways to change this default information:

- A user may override this information using the [obligation](#) transaction.
- If the obligation is in **active** or **pending stop** states, a button appears on [Obligation - Main](#) called **Apply New Start Option**. When pressed, the user is allowed to define a start option and the date its terms become effective on the obligation. Refer to [Changing A Start Option](#) for the details of this functionality.

The topics in this section describe how to setup start options.

**The merge transaction can save setup time.** The Obligation Type Start Options Merge transaction can be used to construct a start option by copying pieces from other start options.

### Contents

- [Obligation Type Start Option - Main](#)
- [Obligation Type Start Option - Rate Info](#)
- [Obligation Type Start Option - Characteristics & Qty](#)
- [Obligation Type Start Option - Terms and Conditions](#)

## Obligation Type Start Option - Main

Open **Admin Menu**, **Obligation Type Start Option** and navigate to the **Main** tab to define an obligation type's start options.

### Description of Page

Every start option is uniquely identified by the following fields:

<b>Division &amp; Obligation Type</b>	Enter the division and obligation type to which the start option is linked.
---------------------------------------	---

**Start Option**

Enter the unique identifier of the option. Pick something easy to recognize as this will be used by CSRs to pick an option when they start service.

**Effective Date**

Enter the earliest effective date. It should be the same as the earliest effective date of the start option's rate (although it doesn't hurt for it to be earlier). The date defaults to the current date. (The status, below, should be **Active**.)

The remaining fields further describe a start option.

Enter a **Description** for the start option.

Indicate its **Status**. For new start options, the status should be **Active**. When it's no longer applicable, change it to **Inactive**.

Enter the **Rate Schedule** that should be defaulted onto obligations created using this option.

**Note.** Only rates that meet the following criteria may be specified: 1) the rate must be defined as valid for the division / obligation type, and 2) the rate must have at least one **Finished** rate version.

For more information about an obligation's rates, refer to [Obligation - Rate Info](#).

Enter the **Recurring Charge Amount** that should be defaulted onto obligations created using this option. This field is only visible when the obligation type allows recurring charges. In addition, the prompt for this field is defined on the obligation type table on the billing window (e.g., it could appear as Payment Amount, Budget Amount, or Installment Amount).

Enter the **Currency Code** in which monetary amounts are denominated.

**Default Note.** The currency code defaults from the [installation record](#).

Enter the **Total Amount to Bill** that should be defaulted onto obligations created using this option. The prompt for this field is defined on the obligation type table on the billing window.

Use **Number of Payment Periods** is not applicable in this release.

If your obligation type has a special role of **Billable Charge**, then you can setup the start option to automatically create a billable charge when an obligation is created using this start option. To use this feature you should turn on the **Create Billable Charge** switch and specify the **Billable Charge Template** that will be used to create the billable charge. These fields are only allowed for obligation types with special role of **Billable Charge**.

Refer to [Setting Up Billable Charge Templates](#) for more information about templates.

## Obligation Type Start Option - Rate Info

Open **Admin Menu, Obligation Type Start Option** and navigate to the **Rate Info** tab to define the start option's default values for contract riders and contract values.

### Description of Page

The information in the **Contract Riders** collection defines the contract riders to be defaulted onto obligations created using this start option. The following fields are required for each instance:

<b>Rate Factor</b>	The rate factor defines the type of rider. You may only reference rate factors designated as being applicable for contract riders.
<b>Number of Days</b>	The number of days the rider should be in effect. This value is used by the system to set the stop date on the obligation's contract rider. <u>If the rider has no expiration, set this field to 0.</u> Default note: this field will be set to 0 if left blank.

For more information about a rate's contract riders, refer to [Defining General Rate Factor Information](#). For more information about an obligation's contract riders, refer to [Obligation – Rate Info](#).

The information in the **Contract Values** collection defines the contract values to be defaulted onto obligations created using this start option. The following fields are required for each event:

<b>Rate Factor</b>	The rate factor defines the type of value. You may only reference rate factors designated as allowing values in contract terms.
<b>Number of Days</b>	The number of days the value should be in effect. This value is used by the system to set the stop date on the obligation's contract value. <u>If the value has no expiration, set this field to 0.</u>
<b>Value</b>	The amount of the contract value.

For more information about a rate's contract values, refer to [Defining General Rate Factor Information](#). For more information about an obligation's contract values, refer to [Obligation – Rate Info](#).

## Obligation Type Start Option - Characteristics & Qty

Open **Admin Menu, Obligation Type Start Option** and navigate to the **Characteristics & Qty** tab to define the start option's default values for characteristics and contract quantities.

### Description of Page

The information in the **Characteristics** collection defines the characteristics to be defaulted onto obligations created using this start option. The following fields are required for each instance:

<b>Characteristic Type</b>	This defines the type of characteristic. Note: You may only define characteristics valid on obligations.
<b>Characteristic Value</b>	This defines the characteristic value that will be defaulted.

For more information about an obligation's characteristics, refer to [Obligation – Chars, Qty & Rec. Charges](#).

The information in the **Contract Quantity** collection defines the contract quantities to be defaulted onto obligations created using this start option. The following fields are required for each instance:

<b>Contract Quantity Type</b>	This defines the type of contract quantity.
<b>Contract Quantity</b>	The amount of the contract quantity.

For more information about an obligation's contract quantities, refer to [Obligation – Chars, Qty & Rec. Charges](#).

## Obligation Type Start Option - Terms and Conditions

Obligations may contain legal terms and conditions (T&Cs) that govern the agreement the taxpayer has with the tax authority.

The system defaults T&Cs for an obligation from the start option used to initially create it, if applicable. You can change an obligation's T&Cs at will.

Open **Admin Menu, Obligation Type Start Option** and navigate to the **Terms and Conditions** tab to define the start option's default terms and conditions.

### Description of Page

The information in the grid defines the terms and conditions to be defaulted onto obligations created using this start option. The following fields are required for each instance:

<b>Terms and Conditions</b>	This is the code that identifies a term and condition (T&C).
<b>Number of Days</b>	The number of days the T&C should be in effect. This value is used by the system to set the end date on the obligation's T&C. If the T&C has no expiration, set this field to 0. This field is defaulted to 0 if left blank.

## Obligation Type Start Options Merge

Use this page to modify an existing obligation type start option (start option) by copying information from other start options.

**Note.** The target start option must exist prior to using this page. If you are creating a new start option, you must first go to the Start Option page to add the new start option and then navigate to the merge page to copy collection information.

**Duplicate versus Merge.** The Start Option page has [Duplication](#) capability. You would duplicate a start option if you want to a) create a new start option AND b) populate it with all the information from an existing start option. You would use the start option merge page if you want to build a start option using pieces of one or more start options.

## Contents

- [Start Options Merge - Main](#)
- [Start Options Merge - Characteristics and Quantities](#)
- [Start Options Merge - Terms and Conditions](#)

## Start Options Merge - Main

Open **Admin Menu, Obligation Type Start Options Merge** and navigate to the **Main** tab to open this page.

### Description of Page

Select the **Original Start Option**, which is the target for merging the start option collection information.

Select the **Merge From Start Option**, which is your template start option to copy the collections from.

**Note.** You may only copy information from one Merge From start option at a time. If you wish to copy information from more than one start option, select the first Merge From start option, copy the desired records, Save, then select the next Merge From start option.

The left portion of the page will display any existing records in the collections for the original start option. The right portion of the page will display the existing records in the collections for the Merge From start option.

You may use the **Copy All** button to copy all the records in all the collections from the Merge From start option to the Original start option. If you do not choose to copy all, you may copy records individually as described below.

The left portion of the **Contract Riders** collection initially displays existing contract riders linked to the original start option. In the **Merge Type**, you will see the word **Original**, for any of these records. The **Rate Factor** and **Number of Days** for each contract rider are displayed. In the right portion of the collection, the existing records in the merge from start option are displayed initially.

The left portion of the **Contract Values** collection initially displays existing contract values linked to the original start option. In the **Merge Type**, you will see the word **Original**, for any of these records. The **Rate Factor**, **Number of Days** and **Value** for each contract value is displayed. In the right portion of the collection, the existing records in the merge from start option are displayed initially.

The topics, which follow, describe how to perform common maintenance tasks:

### Contents

- [Removing A Row From A Grid](#)
- [Adding A New Row To A Start Option](#)
- [Removing An Uncommitted Row From A Start Option](#)

## Removing A Row From A Grid

If you wish to remove a record linked to the Original start option, click the “-” button to the left of the record.

## Adding A New Row To A Start Option

You may move any of the records from the Merge From start option to the original start option by selecting the left arrow adjacent to the desired row. Once a record is moved it will disappear from the Merge From information and appear in the Original information with the word **Merge** in the Merge Type column.

The screen will redisplay with the characteristic moved to the left portion of the page.

## Removing An Uncommitted Row From A Start Option

If you have copied a row across by mistake, you may remove it by clicking on the right arrow adjacent to the appropriate record.

Refer to [Editable Grid](#) in the system wide standards documentation for more information about adding records to a collection by selecting from a list.

# Start Options Merge - Characteristics and Quantities

Open **Admin Menu, Obligation Type Start Options Merge** and navigate to the **Characteristics and Quantities** tab to copy rows in the characteristic and contract quantity collections.

### Description of Page

The left portion of the **Characteristics** collection initially displays existing characteristics linked to the original start option. In the **Merge Type**, you will see the word **Original**, for any of these records. The **Characteristic Type** and **Characteristic Value** for each characteristic are displayed. In the right portion of the collection, the existing records in the merge from start option are displayed initially.

The left portion of the **Contract Quantity** collection initially displays existing contract quantities linked to the original start option. In the **Merge Type**, you will see the word **Original**, for any of these records. The **Contract Quantity Type** and **Contract Quantity** for each contract quantity are displayed. In the right portion of the collection, the existing records in the merge from start option are displayed initially.

Refer to **Obligation Type Start Options Merge - Main** for more information about how to perform common maintenance tasks for the grids displayed on this tab page.



## Start Options Merge - Terms and Conditions

Open **Admin Menu, Obligation Type Start Options Merge** and navigate to the **Terms and Conditions** tab to copy terms and conditions (T&Cs).

### Description of Page

The left side of the **Terms and Conditions** grid initially displays the T&Cs linked to the original start option. On the right side, the T&Cs linked to the merge from start option are displayed initially.

# Background Processes Addendum

This chapter is an addendum to the general [Defining Background Processes](#) chapter. This addendum describes the background processes that are provided with Oracle Enterprise Taxation Management.

## Contents

- [The System Background Processes](#)
- [Batch Process Dependencies](#)
- [How To Set Up A New Extract Processes](#)
- [The Big Picture of Sample & Submit](#)

## The System Background Processes

---

The topics in this section describe functionality that is common to system background processes.

## Contents

- [Process What's Ready Processes](#)
- [Monitor Processes](#)
- [Extract Processes](#)
- [Adhoc Processes](#)
- [ToDo Entry Processes](#)
- [Object Validation Processes](#)
- [Referential Integrity Validation Processes](#)
- [Conversion Processes](#)
- [Conversion Processes Executed In The Staging Database](#)
- [Purge Processes](#)
- [ConfigLab Processes](#)
- [Archive and Purge Processes](#)
- [Column Descriptions](#)

## Process What's Ready Processes

Some background processes create and update records that are “ready for processing”. The definition of “ready” differs for every process. For example,

- The payment upload process creates payments for every record that is pending
- The overdue event monitor activates pending overdue events that have reached their trigger date

Processes of this type tend to use a business date in their determination of what's ready. If the requester of the process does not supply a specific business date, the system assumes that the current system date should be used. If you need to use a date other than the current date, simply supply the desired date when you request the batch process.

The following table lists every background process that processes all data that is “ready”.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
ACTVTAPY	CIPPAAPB	<p>This process marks each auto pay download staging record with the batch control associated with its auto-pay source's route type. It also stamps the respective batch control's current run number on each record.</p> <p>Note: The APAYACH background process uses the information on this staging table to create the flat file that is used to interface information to the ACH. The BALAPY background process uses the information on this staging table to create automatic payment tender controls.</p> <p>Refer to <a href="#">Activating Automatic Payments</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	200/15
APAYCRET	CIPPACRB	<p>This process creates automatic payments for bills whose automatic payment creation has been deferred until the extract date. This extract date is stamped on the bill and is used by this background process to select all bills whose automatic payment extract date is on or before the supplied business date. It calls the automatic payment creation algorithm plugged in on the installation record to create the automatic payments. Note that the algorithm supplied does not distribute and freeze the automatic payments that are created. This is handled by the complementary background process <a href="#">APAYDSFR</a>.</p> <p>Refer to <a href="#">Installation Options - Billing and Automatic Payments</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	300/15
APAYDSFR	CIPPADFB	<p>This process distributes and freezes automatic payments whose distribution date (indicated on the download staging record) is on or before the supplied business date. Payments that have been distributed (e.g., manually) are frozen if the above criterion is satisfied.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	300/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
		<p>This job complements the <a href="#">APAYCRET</a> background process and the <a href="#">PPAPAY</a> background process when the <b>Autopay Creation Option</b> on the installation record is set to <a href="#">Create on Extract Date</a>.</p> <p>Refer to <a href="#">Installation Options - Billing</a> and <a href="#">Automatic Payments</a> for more information.</p>				
BALAPY	CIPFBAPB	<p>This process creates a new tender control (with an associated deposit control) for each batch control and run number encountered for extracted automatic payments that are not already linked to a tender control.</p> <p>Afterwards, this process balances the open tender and deposit control records.</p> <p>Note: Automatic payment staging records are activated by the ACTVTAPY process and extracted by the APAYACH.</p> <p>Refer to <a href="#">Creating Automatic Payment Tender Controls</a> for more information.</p>	No	<a href="#">MAX-ERRORS</a>	Yes	200/15
BCASSIGN	CIPFBCAB	<p>This process assigns the <a href="#">Pending</a> balance control group to new FT's (i.e., those without a balance control group).</p> <p>Refer to <a href="#">The Big Picture of Balance Control</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	No	200/15
BCGNEW	CIPFBCGB	<p>This process creates a <a href="#">Pending</a> balance control group if one doesn't already exist.</p> <p>Refer to <a href="#">The Big Picture of Balance Control</a> for more information.</p>	No	<a href="#">MAX-ERRORS</a>	No	N/A (only 1 record is inserted)
BCGSNAP	CIPFBCSB	<p>The balance control snapshot and verification process has two functions:</p> <ol style="list-style-type: none"> <li>1. It summarizes the number and value of the financial transactions on the current <a href="#">Pending</a> balance control group record.</li> <li>2. It verifies the financial integrity of your system.</li> </ol> <p>The value of the VERIFY-ONLY-SW parameter controls which of these</p>	No	VERIFY-ONLY-SW <a href="#">MAX-ERRORS</a>	No	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
		<p>functions is performed:</p> <ul style="list-style-type: none"> <li>- If VERIFY-ONLY-SW = "N", the system summarizes the new financial transactions under the current <i>Pending</i> balance control and verifies that the balances summarized on <u>every</u> historical balance control group are consistent with the financial transactions associated with this balance control group (i.e., it checks the financial integrity of the system).</li> <li>- If VERIFY-ONLY-SW = "G", the system only summarizes the new financial transactions under the current <i>Pending</i> balance control (i.e., the verification step is not performed).</li> <li>- If VERIFY-ONLY-SW = "Y", the system verifies that the balances summarized on <u>every</u> historical balance control group are consistent with the financial transactions associated with this balance control group (i.e., it checks the financial integrity of the system).</li> </ul> <p>Note: You may want to use the VERIFY-ONLY-SW parameter to improve system performance. For example, you can generate the balance control summary nightly (run the process with the switch set to "G") and validate the balance control summaries weekly (run the process with the switch set to "Y").</p> <p>Refer to <a href="#">The Big Picture of Balance Control</a> for more information.</p>				
BCU1	CIPCBC1B	<p>The first phase of the billable charge upload staging process validates and defaults information on to billable charge upload staging records.</p> <p>Refer to <a href="#">Billable Charge Upload Background Processes</a> for more information.</p>	No	<a href="#">MAX-ERRORS</a>	No	N/A
BCU2	CIPCBC2B	The second phase of the billable charge	Yes	<a href="#">MAX-ERRORS</a>	No	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
		upload staging process creates billable charges for the new billable charge upload staging records.  Refer to <a href="#">Billable Charge Upload Background Processes</a> for more information.				
BILLING	CIPBBILB	The bill cycle process creates bills for accounts with an "open" bill cycle.  Refer to <a href="#">Batch Billing</a> for more information.	Yes	<a href="#">MAX-ERRORS</a>	Yes	100/15
C1-ADMOV	CIPLOVMB	The overdue monitor uses your overdue rules to collect overdue debt. Refer to <a href="#">How Does The Overdue Monitor Work?</a> for more information.	Yes	<a href="#">MAX-ERRORS</a>	Yes	2000/15
C1-APYDF	Java	This process distributes and freezes automatic payments using payment distribution rules.  The background process reads all the automatic payment records whose scheduled distribution date is on or before the business date. For each record, payment(s) and pay segment(s) are created using the payment event's distribution rule(s). Payments are frozen.	Yes	<a href="#">MAX-ERRORS</a>	Yes	200/NA
C1-CALPI	Java	This process brings P&I up to date to the input business date for all obligations.  Refer to <a href="#">The Big Picture of Penalty and Interest</a> for more information.	Yes	<a href="#">MAX-ERRORS</a>	Yes	200/NA
C1-CSTRS	CIPQTRCB	The case scheduled transition process transitions cases to a nominated <b>next status</b> or <b>transition condition</b> at a scheduled time.  The process selects all open cases whose current status is linked to the process' batch control code and are allowed to transition from their current status to the chosen next status or condition (i.e. where	Yes	NEXT-STATUS-CD (Next Status Code) NEXT-TR-COND-FLG (Next Transition Condition) <a href="#">MAX-ERRORS</a>	Yes	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
		a corresponding transition rule exists for the case type/status combination) based on the input algorithm parameters.				
C1-ODET	CIPLOETB	<p>The overdue / cut event manager activates all overdue and cut events whose trigger date is on or before the supplied business date. Refer to <a href="#">How and When Events Are Activated</a> for more information. This process also has the responsibility of recursively activating later events that are dependent on the completion of earlier events.</p> <p>For overdue or cut events that are in the <i>Wait</i> state, this process runs the associated waiting algorithm for the event type to determine if the object the event is waiting for is complete (and then triggering the dependent events when it completes).</p> <p>Populate an Overdue Process Template in the input parameter to limit the processing to overdue processes for this template.</p>	Yes	OD-PROC-TMP-CD (Optional) <a href="#">MAX-ERRORS</a>	Yes	2000/15
C1-PEPL1	CIPPEL1B	<p>This is the first of three background processes that load the contents of the payment event upload staging records into the various payment tables.</p> <p>It first creates new deposit and tender control records, and then updates the payment event upload staging records with the corresponding Tender Control ID.</p> <p>Next, it processes each <i>incomplete</i> record as follows: It updates the record's Tender Account ID with the account ID returned by the <i>Determine Tender Account</i> algorithm defined on the <a href="#">distribution rule</a>. If the Pay Event Process ID field is not populated, it is set equal to the tender account ID. If no error was encountered, it transitions the record from to <i>Pending</i>.</p> <p>Refer to <a href="#">Interfacing Payments Using Distribution Rules</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	300/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
C1-PEPL2	CIPPEL2B	<p>This is the second of three background processes that load the contents of the payment event upload staging records into the various payment tables.</p> <p>The responsibility of this process is to create payment events, payment tenders and payments and transition the corresponding staging records from <i>Pending</i> to <i>Complete</i>.</p> <p>Refer to <a href="#">Interfacing Payments Using Distribution Rules</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	300/15
C1-PEPL3	CIPPEL3B	<p>This is the last of three background processes that load the contents of the payment event upload staging records into the various payment tables.</p> <p>The responsibility of this process is to update the status of the related deposit and tender controls from <i>open</i> to <i>balanced</i>.</p> <p>Refer to <a href="#">Interfacing Payments Using Distribution Rules</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	No	300/15
CASETRAN	CIPOCSTB	<p>This batch process is responsible for calling the algorithm that determines if a case should be transitioned to a new state. Refer to <a href="#">Automatic Transition Rules</a> for the details.</p> <p>If <b>Restrict To Case Type Code</b> is specified, only cases of this type will be analyzed to determine if they should be transitioned to a new state.</p> <p>If <b>Restrict To Status Code</b> is specified, only cases in this status will be analyzed to determine if they should be transitioned to a new state. Note, if this parameter is specified, a <b>Restrict To Case Type Code</b> must also be defined.</p>	Yes	CASE-TYPE-CD (Restrict To Case Type Code)  CASE-STATUS-CD (Restrict To Status Code)  <a href="#">MAX-ERRORS</a>	Yes	200/15
GLASSIGN	CIPFGLAB	The GL account number assignment process assigns GL account numbers to the GL details associated with financial	Yes	<a href="#">MAX-ERRORS</a>	Yes	200/15



Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
		transactions. Refer to <a href="#">The GL Interface</a> for more information.				
GLS	CIPFGLEB	<p>The create general ledger download staging process creates a download staging record for every financial transaction that is ready for download.</p> <p>This process populates the FT / Batch Process table with the unique ID of all financial transactions to be interfaced to the general ledger. This process marks each staging record with the GL interface's batch process ID (defined on the installation record). It also stamps the respective batch control's current run number on each record.</p> <p>Note: The GLDL background process uses the information on this staging table to create the consolidated journal entries that are interfaced to your general ledger.</p> <p>Refer to <a href="#">The GL Interface</a> for more information.</p>	Yes	<a href="#">MAX-ERRORS</a>	No	200/15
C1-PAYPA	CIPGACRB	<p>The Pay Plan scheduled payment autopay create process creates autopay records for any scheduled pay plan payments where the account is set up for autopay and the pay plan is not excluded from autopay.</p> <p>For more information about pay plan auto payments, refer to <a href="#">Automatic Payment and Pay Plans</a>.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	300/15
C1-PAYPS	CIPGNPSB	<p>The pay plan scheduled payment process performs processing for scheduled payment records with a payment date on or before the process business date.</p> <p>After processing, the scheduled payment status is updated to processed.</p> <p>For more information, refer to <a href="#">Processing Scheduled Payments</a>.</p>	Yes	<a href="#">MAX-ERRORS</a>	Yes	300/15
C1-RDAMT	CIPFFTRB	This process looks for financial transactions linked to each obligation that	Yes	<a href="#">MAX-ERRORS</a>	Yes	100/10

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minute Between Cursor Re-Initiation
		are older than X days (where X is defined on the installation record) that sum to zero. If it finds such FTs, it marks them as "redundant". Redundant FTs do not have to be accessed by the various SQL statements that accumulate an account or obligation's balance.				
PUPL	CIPPUPLB	The upload payments process creates payment events, payments, and tenders using the records in the various payment staging tables.  Refer to <a href="#">Interfacing Payments From External Sources</a> for more information.	Yes	<a href="#">MAX-ERRORS</a>	No	100/15
PY-RPE	CIPPRPEB	The resolve payments in error process attempts to resolve the following payment errors automatically: <ul style="list-style-type: none"> <li>A valid account was found but no active obligation exists.</li> </ul> Refer to <a href="#">Resolving Exceptions Automatically</a> for more information.	Yes	<a href="#">MAX-ERRORS</a>	No	200/15
REDSAAMT	CIPFFTRB	This process looks for financial transactions linked to each obligation that are older than X days (where X is defined on the installation record) that sum to zero. If it finds such FTs, it marks them as "redundant". Redundant FTs do not have to be accessed by the various SQL statements that accumulate an account or obligation's balance.	Yes	<a href="#">MAX-ERRORS</a>	Yes	100/10
SAACT	CIPCSATB	The obligation activation process updates pending start and pending stop obligations.	Yes	<a href="#">MAX-ERRORS</a>	Yes	200/15

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## Monitor Processes

A periodic monitor batch process is provided for any maintenance object whose business object defines a [lifecycle](#). In addition deferred monitor batch process is provided if a business object supplied in the base product required a deferred process for one of its states.

Refer to [Monitoring Batch Processes](#) for more information.

## Extract Processes

Extract processes extract information that is interfaced out of the system. Processes of this type typically extract records marked with a given run number. If the requester of the process does not supply a specific run number, the system assumes that the latest run number should be extracted. If you need to re-extract an historical batch, you can – simply supply the respective run number when you request the batch process.

**Business Intelligence Extracts.** If you are using Oracle Utilities Business Intelligence, there will be many other extract processes. These additional processes are responsible for extracting the data sent to the data-warehouse. Please see the product's fact and dimension chapter of the business intelligence documentation for a description of each extract process.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits. Minutes Between Cursor Re Initiation
APAYACH	CIPPXAPB	<p>The automatic payment ACH (automated clearing house) download extraction process creates the flat file that is interfaced to the ACH. This process downloads all auto pay download staging records associated with its batch control ID that are marked with a supplied run number. If a run number is not supplied, the process extracts all automatic payment download records marked with the current run number.</p> <p>Note: the ACTVTAPY process updates auto pay download records on their extract date so that they will be downloaded by this process.</p> <p>Refer to <a href="#">Downloading Automatic Payments</a> for more information.</p>	Yes	<p><a href="#">FILE-PATH</a>= directory path into which output should be placed</p> <p><a href="#">FILE-NAME</a>= name of file into which output should be placed</p> <p><a href="#">MAX-ERRORS</a></p>	NA
APDL	CIPADAPB	<p>The A/P download process creates the flat file that is interfaced to your accounts payable software (to cut checks).</p> <p>The process that is delivered has skeletal logic and must be customized by your organization to satisfy the needs of your accounts payable software.</p> <p>In order to adapt the base product program to your specific needs, please following the standard steps:</p> <ul style="list-style-type: none"> <li>Copy the base product program to your own program. Your own program should be prefixed with the letters <b>CM</b> (which stands for "taxpayer modification"). This is important as it prevents the</li> </ul>	Yes	<p><a href="#">FILE-PATH</a>= directory path into which output should be placed</p> <p><a href="#">FILE-NAME</a>= name of file into which output should be placed</p> <p><a href="#">MAX-ERRORS</a></p>	NA

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
		<p>upgrade process from overwriting your new logic.</p> <ul style="list-style-type: none"> <li>Introduce logic to format the downloaded records into your specific format.</li> <li>If you need assistance, please contact the implementation support group.</li> </ul> <p>This process uses all adjustment extract records associated with its batch control that are marked with a supplied run number. If a run number is not supplied, the process uses all A/P request extract records marked with the current run number.</p> <p>Refer to <a href="#">The A/P Interface</a> for more information.</p>			
DWLDCOLL	CIPLXCRB	<p>The collection agency referral download extraction process creates the flat file that contains referrals to be interfaced to your collection agencies. This process extracts all collection agency referral history records associated with its batch control that are marked with a supplied run number. If a run number is not supplied, the process extracts all referral history records marked with the current run number.</p> <p>The program that is delivered contains skeletal logic that should be used as the basis for your specific processing. The skeletal logic does NOT extract information to a flat file; you must introduce the logic to support your specific flat file format.</p> <p>In order to adapt the base product program to your specific needs, please following the standard steps:</p> <ul style="list-style-type: none"> <li>Copy the base product program to your own program. Your own program should be prefixed with the letters <b>CM</b> (which stands for "taxpayer modification"). This is important as it prevents the upgrade process from overwriting your new logic.</li> <li>Introduce logic to format the downloaded records into your specific format.</li> </ul> <p>Note: records are written to the referral history table when a collection agency oriented write-off events are activated. Referral history records may also be added manually by an operator.</p> <p>Refer to <a href="#">The Big Picture of Collection Agency Referrals</a> for more information.</p>	No	<p><a href="#">FILE-PATH</a>= directory path into which output should be placed</p> <p><a href="#">FILE-NAME</a>= name of file into which output should be placed</p> <p><a href="#">MAX-ERRORS</a></p>	NA

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
GLDL	CIPFXGLB	<p>The general ledger download process creates the flat file that is interfaced to your general ledger software.</p> <p>This process uses all FT / Batch Process records associated with its batch control that are marked with a supplied run number. If a run number is not supplied, the process uses all FT / Process records marked with the current run number.</p> <p>In order to adapt the base product program to your specific needs, please following the standard steps:</p> <ul style="list-style-type: none"> <li>Copy the base product program to your own program. Your own program should be prefixed with the letters <b>CM</b> (which stands for "taxpayer modification"). This is important as it prevents the upgrade process from overwriting your new logic.</li> <li>Introduce logic to format the downloaded records into your specific format.</li> </ul> <p>Refer to <a href="#">The GL Interface</a> for more information.</p>	No	<p><a href="#">FILE-PATH</a>= directory path into which output should be placed</p> <p>FILE-NAME= name of file into which output should be placed</p> <p><a href="#">MAX-ERRORS</a></p>	NA
LTRPRT	CIPCLTPB	<p>The customer contact letter download process creates the flat file(s) that are interfaced to your letter print software to print letters associated with letter-oriented customer contacts.</p> <p>This process extracts all customer contact records associated with its batch control ID that are marked with a supplied run number. If a run number is not supplied, the process uses all customer contact records associated with its batch control ID that are marked with the current run number.</p> <p>Each downloaded letter's output is written to a filename that is a concatenation of the letter's Letter Template Code and the process's Thread Number. This means that this process can write to multiple files as multiple Letter Template Codes may be downloaded by this process.</p> <p>The information that is extracted and placed on the flat file for each letter is controlled by each customer contact's letter template's extract algorithm. Refer to <a href="#">Letter Templates Control The Information Merged Onto Letters</a> for information about how a letter's flat file records are constructed.</p> <p>The <b>FILE-PATH</b> parameter controls where the output</p>	Yes	<p><a href="#">FILE-PATH</a>= directory path into which output should be placed</p> <p>FIELD-DELIM-SW=Y or N</p> <p>CNTL-REC-SW=Y or N</p> <p><a href="#">MAX-ERRORS</a></p>	NA

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re-Initiation
		<p>files are placed.</p> <p>The format of the information on the flat file can be either tilde delimited or in a fixed position (based on the <b>FIELD-DELIM-SW</b> parameter). Tilde delimited output is used if you merge the information into a Word template. Fixed position output is used if you merge the information into a template that expects this type of format.</p> <p>You can use the <b>CNTL-REC-SW</b> parameter to cause the extract to produce a control record that contains batch code, run number, number of letters to print, etc.</p> <p>Refer to <a href="#">Printing Letters</a> for more information.</p>			
POSTROUT	CIPBXLB	<p>The bill print process creates the flat file that is interfaced to your bill print software. This process uses all bill routing extract records associated with its batch control that are marked with a supplied run number. If a run number is not supplied, the process extracts all bill routing extract records marked with the current run number.</p> <p>The information that is extracted and placed on the flat file for each bill is controlled by each bill route type's extract algorithm. Refer to <a href="#">Bill Route Controls The Information Merged Onto Bills</a> for information about how a bill's flat file records are constructed.</p> <p>The <b>FILE-PATH</b> parameter controls where the output files are placed.</p> <p>Refer to <a href="#">Printing Bills</a> for more information.</p>	Yes	<p><a href="#">FILE-PATH</a>= directory path into which output should be placed</p> <p><b>FILE-NAME</b>= name of file into which output should be placed</p> <p><a href="#">MAX-ERRORS</a></p>	NA

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## Adhoc Processes

These are background processes that are run on an ad hoc basis (e.g., if you need to back out bills that were created by the billing process).

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commits Minutes Between Cursor Re-Initiation

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minutes Between Cursor Re-Initiation
F1-AVALG	Java	This process regenerates algorithm type and their related algorithm information to be displayed by the Application Viewer. It produces a series of XML files in a designated folder under the application viewer /data folder.  Refer to <a href="#">Application Viewer Generation</a> for more information on this background process is used.	No	<a href="#">MAX-ERRORS</a>	No	NA
F1-AVMO	Java	This process regenerates maintenance object information to be displayed by the Application Viewer. It reads the meta-data maintenance object information and produces a series of XML files in a designated folder under the application viewer /data folder.  Refer to <a href="#">Application Viewer Generation</a> for more information on this background process is used.	No	<a href="#">MAX-ERRORS</a>	No	NA
F1-AVTBL	Java	This process regenerates table and column information to be displayed by the Application Viewer. It reads the meta-data table and related entities and produces a series of XML files in a designated folder under the application viewer /data folder.  Refer to <a href="#">Application Viewer Generation</a> for more information on this background process is used.	No	<a href="#">MAX-ERRORS</a>	No	NA
MASSCNCL	CIPBMCNB	The mass bill cancellation process removes an entire batch of bills that were created by the BILLING process.  Refer to <a href="#">Canceling A Batch Of Bills After They're Complete</a> for more information.	Yes	BILL-CYC-CD= the bill cycle associated with the bills  WIN-START-DT=the bill cycle window start date associated with the bills. This should be entered in the format YYYY-MM-DD.  BILL-DT= the date on which the bills were created. This	Yes	100/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commit Minutes Between Cursor Re-Initiation
				should be entered in the format YYYY-MM-DD. <a href="#">MAX-ERRORS</a>		
MASSROBL	CIPBMROB	The mass bill reopen process reopens an entire batch of bills that were completed by the BILLING process.  Refer to <a href="#">Reopening A Batch Of Bills After They're Complete</a> for more information.	Yes	BILL-CYC-CD= the bill cycle associated with the bills  WIN-START-DT= the bill cycle window start date associated with the bills  BILL-DT= the date on which the bills were created. This should be entered in the format YYYY-MM-DD.  <a href="#">MAX-ERRORS</a>	Yes	200/15
NEWLANG	CIPZLNGB	This Create New Language batch program duplicates all language specific rows from the source to the target language. Once this program has run you will need to translate the language specific columns.  This program can be rerun at anytime. It will only duplicate entries where they do not already exist.  Note if you run this program with the "DELETE" action then the source and target language must be the same.  If you have any questions, please see your implementation team for more information.	Yes	SOURCE-LANGUAGE= source language code  TARGET-LANGUAGE= target language code  PROGRAM-ACTION= (DELETE or INSERT)  <a href="#">MAX-ERRORS</a>	No	200/15
UPDERR	CIPZUESB	The process updates <i>In Progress</i> threads in an abnormally terminated batch run to be <i>Error</i> . When at least one thread is in <i>Error</i> , this process also updates the status of the batch run to be <i>Error</i> .  Refer to <a href="#">Dealing With Abnormally Terminated Background Processes</a> for information about when and why this process would be	No	BATCH-CD-IN-PROGRESS = the batch control ID of the abnormally terminated batch process  BATCH-NBR-IN-PROGRESS = the	No	N/A



Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Error Generates To Do	Record Between Commits Minutes Between Cursor Re-Initiation
		executed.		<p>run number of the abnormally terminated batch process</p> <p>UPD-ALL-THRDS-SW must be Y or N. A value of Y means that the status of all <i>In Progress</i> threads will be changed to <i>Error</i>. A value of N means that the next parameter must be supplied.</p> <p>BATCH-THRD-NBR-IN-PROGRESS = the thread number whose status should be changed from <i>In Progress</i> to <i>Error</i>. This parameter should only be specified if UPD-ALL-THRDS-SW = N.</p> <p><a href="#">MAX-ERRORS</a></p>		

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## ToDo Entry Processes

These are background processes whose main purpose is to generate To Do Entry records based on a certain condition. Refer to [ToDo Entries Created By Background Processes](#) for the details. The section that appears below simply lists these processes.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re-Initiation

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re-Initiation
TD-BCUPL	CIQOBCEB	This background process creates a To Do entry for every billable charge upload record that's in error.	No	<a href="#">MAX-ERRORS</a>	200/15
TD-BIERR	CIQOBIEB	This background process creates a To Do entry for every bill that's in error.	Yes	<a href="#">MAX-ERRORS</a>	200/15
TD-BSERR	CIQOBSEB	This background process creates a To Do entry for every bill segment that's in error.	Yes	<a href="#">MAX-ERRORS</a>	200/15
TD-BTERR	CIQOBERB	This background process creates To Do Entry for any other batch processes that ended in error. A To Do Entry is only created if one does not already exist.	No	<a href="#">MAX-ERRORS</a>	200/15
TD-CCCB	CIQOCCCB	This background process creates a To Do entry for customer contacts that have been flagged to generate a To Do entry on a future date. Note well, most To Do background processes create To Do entries in the <i>pending</i> state. If the customer contact indicates a specific user should be notified (as opposed to notifying a group of users – a role), the To Do entry will be created in the <i>being worked</i> state and it will be assigned to the designated user.	Yes	LEAD-DAYS = Number of days before the customer contact's reminder date that the To Do entry should be created. Valid values of 0 to 99 are acceptable. <a href="#">MAX-ERRORS</a>	200/15
TD-CLERR	CIQOCLEB	This background process creates a To Do Entry for any batch process that has root objects that created an error. A To Do Entry is only created if one does not already exist.	Yes	<a href="#">MAX-ERRORS</a>	200/15
TD-DTCST	CIQODTCB	This background process creates a To Do entry for deposit control staging / tender control staging records that are in error.	No	<a href="#">MAX-ERRORS</a>	200/15
TD-MODTL	CIQOODMB	This background process creates a To Do entry for every disputed match event	Yes	NO-OF-DAYS = Number of days old the match event must be before a To Do entry is created (this prevents young entries from appearing on To Do lists) <a href="#">MAX-ERRORS</a>	200/15
TD-	CIQOONMB	This background process creates a To Do entry for	Yes	NO-OF-DAYS =	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor R Initiatio
MONTL		every open, non-disputed match event		Number of days old the match event must be before a To Do entry is created (this prevents young entries from appearing on To Do lists) <a href="#">MAX-ERRORS</a>	
TD-NOBC	CIQNBCB	This background process creates a To Do entry for every account that doesn't have a bill cycle but has active obligations.	Yes	<a href="#">MAX-ERRORS</a>	200/15
TD-PYERR	CIQPAYB	This background process creates a To Do entry for every payment that's in error or that is unfrozen.	Yes	<a href="#">MAX-ERRORS</a>	200/15
TD-PYUPL	CIQPYUB	This background process creates a To Do entry for every payment staging record that's in error.	No	<a href="#">MAX-ERRORS</a>	200/15
TD-UNBAL	CIQPYEB	This background process creates a To Do entry for every payment event that's unbalanced.	Yes	<a href="#">MAX-ERRORS</a>	200/15
TD-XAIUP	CIQXAUB	This background process creates a To Do entry for every XAI Upload Staging in error.	Yes	<a href="#">MAX-ERRORS</a>	200/15

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## Object Validation Processes

These background processes are run to validate the master data objects. These programs are typically only run as part of the conversion and upgrade processes.

**Another use for these programs.** In addition to validating your objects after conversion or an upgrade, the validation programs listed below have another use. For example, you want to experiment with changing the validation of a person and you want to determine the impact of this new validation on your existing persons. You could change the validation and then run the person validation object – it will produce errors for each person that fails the new validation.

Refer to [Validate Information In The Staging Tables](#) for more information about these processes and where their errors appear.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
VAL-ACCT	CIPVACCB	Validate the account object	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row. <a href="#">MAX-ERRORS</a>	200/15
VAL-BCHG	CIPVBCGB	Validate the billable charge object	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row. STATUS1=validate rows with this status STATUS2=validate rows with this status <a href="#">MAX-ERRORS</a>	200/15
VAL-CLCS	CIPVCCSB	Validate collection case	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
				example 10 to process every 10 <sup>th</sup> row.	
VAL-OBLG	CIPVSVAB	Validate the obligation object	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row. <a href="#">MAX-ERRORS</a>	200/15
VAL-OVPY	CIPVOPPB	Validate overpayment process object	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value	200/15
VAL-PER	CIPVPERB	Validate the person object	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row. <a href="#">MAX-ERRORS</a>	200/15
VAL-PREM	CIPVPRMB	Validate the location object	Yes	OVRD-LOW-ID=key value to override the calculated start-key value	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
				OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row. <a href="#">MAX-ERRORS</a>	
VAL-TAXR	CIPVTXRB	Validate tax role	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row.	200/15
VAL-TXFR	CIPVTXFB	Validate tax form	Yes	OVRD-LOW-ID=key value to override the calculated start-key value OVRD-HIGH-ID=key value to override the calculated end-key value SKIP-ROWS= nth row to be processed, for example 10 to process every 10 <sup>th</sup> row.	200/15

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## Referential Integrity Validation Processes

The following table lists every background process that validates transaction data using the same program **CIPVRNVB**. These programs are typically run as part of the conversion and upgrade processes.

In all cases, the processes are not multi-threaded and do not include extra parameters. In addition, Records Between Commits and Minutes Between Cursor Re-Initiation are not applicable.

Refer to [Validate Information In The Staging Tables](#) for more information about these processes and where their errors appear.

Batch Control ID	Description
CIPVAAPV	Foreign Key validation for Account Automatic Payment
CIPVACHV	Foreign Key validation for Account Characteristics
CIPVACPV	Foreign Key validation for Account Person Relationship
CIPVADJV	Foreign Key validation for Adjustment
CIPVAPAV	Foreign Key validation for Location Alternate Address
CIPVAPCV	Foreign Key validation for Account Person Characteristics
CIPVAPRV	Foreign Key validation for A/P Check Request
CIPVARHV	Foreign Key validation for Collection Agency Referral History
CIPVARSV	Foreign Key validation for Credit Review Schedule
CIPVBCHV	Foreign Key validation for Bill Characteristic
CIPVBCGV	Foreign Key validation for Billable Charge
CIPVBCLV	Foreign Key validation for Billable Charge Line
CIPVBFVV	Foreign Key validation for Rate Factor Value
CIPVBLLV	Foreign Key validation for Bill Header
CIPVBLMV	Foreign Key validation for Bill Messages
CIPVBLRV	Foreign Key validation for Bill Routing
CIPVBSAV	Foreign Key validation for Bill - Obligation Balance Snapshot
CIPVBSCV	Foreign Key validation for Bill Segment Calc Header
CIPVBSLV	Foreign Key validation for Bill Segment Calc Line
CIPVCARV	Foreign Key validation for Collection Agency Referral
CIPVCCLV	Foreign Key Validation for Collection Case Log
CIPVCCOV	Foreign Key Validation for Collection Case Overdue Process
CIPVCCPV	Foreign Key Validation for Collection Case Log Parameter
CIPVCCSV	Foreign Key Validation for Collection Case
CIPVCRTV	Foreign Key validation for Credit Rating History
CIPVCSCV	Foreign Key validation for Customer Contact
CIPVFTFV	Foreign Key validation for Financial Transaction
CIPVFTGV	Foreign Key validation for Financial Transaction Gen Ledger
CIPVFTPV	Foreign Key validation for Financial Transaction Process
CIPVFTCV	Foreign Key validation for Financial Transaction Characteristics

Batch Control ID	Description
CIPVMSGV	Foreign Key validation for Account Bill Messages
CIPVNBSV	Foreign Key validation for Pay Plan / Obligation
CIPVNPMV	Foreign Key validation for Pay Plan Payment Schedule Parameter Values
CIPVNSPV	Foreign Key validation for Pay Plan Scheduled Payments
CIPVOPCV	Foreign Key validation for Overpayment Process Characteristics
CIPVOPLV	Foreign Key validation for Overpayment Process Log
CIPVOPMV	Foreign Key validation for Overpayment Process Log Parameter
CIPVOPPV	Foreign Key validation for Overpayment Process
CIPVPAOV	Foreign Key validation for Person Address Override
CIPVPAYV	Foreign Key validation for Payment Header
CIPVPYCV	Foreign Key validation for Payment Characteristic
CIPVPCHV	Foreign Key validation for Location Characteristic
CIPVPGOV	Foreign Key validation for Location Geographic location
CIPVPIDV	Foreign Key validation for Person Identifier
CIPVPNMV	Foreign Key validation for Person Name
CIPVPPEV	Foreign Key validation for Person to Person
CIPVPPHV	Foreign Key validation for Person Phone
CIPVPRCV	Foreign Key validation for Person Characteristics
CIPVPSAV	Foreign Key validation for Person Seasonal Address
CIPVPSGV	Foreign Key validation for Payment Segment
CIPVSACV	Foreign Key validation for Obligation Characteristics
CIPVSAHV	Foreign Key validation for Obligation Rate Schedule History
CIPVSAOV	Foreign Key validation for Obligation Contract Terms
CIPVSAQV	Foreign Key validation for Obligation Contract Quantity
CIPVSARV	Foreign Key validation for Obligation Recurring Charge
CIPVSEGV	Foreign Key validation for Bill Segment
CIPVSMGV	Foreign Key validation for Obligation Message
CIPVSQTV	Foreign Key validation for Bill Segment Rate Quantity
CIPVSRRV	Foreign Key validation for Bill Segment Register Read
CIPVTFCV	Foreign Key Validation for Tax Form Characteristics
CIPVTFLV	Foreign Key Validation for Tax Form Log
CIPVTLPV	Foreign Key Validation for Tax Form Log Parameter
CIPVTNDV	Foreign Key validation for Payment Tender
CIPVTNCV	Foreign Key validation for Payment Tender Characteristics
CIPVTXCV	Foreign Key validation for Tax Role Characteristics



Batch Control ID	Description
CIPVTXFV	Foreign Key validation for Tax Form
CIPVTXRV	Foreign Key validation for Tax Role

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## Conversion Processes

These background processes are run only when converting or migrating data from external applications into the system. Your company may never use them depending upon your data migration strategy.

Refer to [The Conversion Tool](#) for more information about these processes and where their errors appear.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
CNV-ADM	CIPVADMB	Creates ADM triggers for converted accounts.	Yes	<a href="#">MAX-ERRORS</a>	200/15
CNV-BCG	CIPVCBCB	This process resets the Balance Control column on all FT's so that the FT's can be included in a balance control (see the last step below) after they have been transferred to production.	Yes	<a href="#">MAX-ERRORS</a>	200/15

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## Conversion Processes Executed In The Staging Database

There are many other background processes that are only executed if you use the conversion tool to load historical data into your production database. These programs perform the following tasks:

- **Key Assignment Programs.** Background processes of this type assign random, clustered keys to the rows in the staging database.

A separate background process exists for every table with a system-assigned key that is supported by the conversion tool. The program names of these processes are documented in [The Conversion Tool](#) (scan for all references to "Key Assignment Program" for a matrix containing these program names).

- **Insertion Programs.** Background processes of this type insert converted rows into production from the staging database.

A separate background process exists for every table that is supported by the conversion tool. The program names of these processes are documented in [The Conversion Tool](#) (scan for all references to “Insertion Program” for a matrix containing these program names).

## Purge Processes

These background processes are used to purge historical records from certain objects that generate a large number of entries and may become unwieldy over time.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits / Minute Between Cursor Re-Initiation
BCUP-PRG	CIPCDBSB	Purges <i>completed</i> billable charge upload objects.	Yes	NO-OF-DAYS = number of days after the creation date that a <i>completed</i> billable charge upload object should be purged <a href="#">MAX-ERRORS</a>	200/15
PYUP-PRG	CIPPDUSB	Purges <i>completed</i> tender upload objects.	Yes	NO-OF-DAYS = number of days after the related tender's payment event's creation date that a <i>completed</i> tender upload object should be purged <a href="#">MAX-ERRORS</a>	200/15
TD-PURGE	CIPQDTDB	Purges <i>completed</i> To Do entries.	Yes	NO-OF-DAYS = number of days after the completion date that a <i>completed</i> To Do entry should be purged DEL-ALL-TD-SW = Y or N. If this switch is Y, all <i>completed</i> To Do entries that are old enough will be deleted. If N, the next parameter defines the specific type of To Do entry that will be deleted. DEL-TD-TYPE-CD. This parameter is only used if DEL-ALL-TD-SW is N. It contains the To Do type code whose <i>completed</i> entries will be deleted. <a href="#">MAX-ERRORS</a>	200/15
XMLUP-PR	CIPXDXUB	Purges <i>completed</i> XML upload objects.	Yes	NO-OF-DAYS = number of days after the completion date that a <i>completed</i> XML upload object should be purged <a href="#">MAX-ERRORS</a>	200/15

Please refer to [Column Descriptions](#) for more information on the columns used in the table above.

## ConfigLab Processes

The following table lists system background processes used in conjunction with the [Configuration Lab](#). The delivered system background processes are designed to copy sample DB processes from the demonstration database.

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
CL-APPCH	CIPYUPDB	The apply changes based on compare is used to update the current environment based on the results of a compare from a <a href="#">ConfigLab</a> or <a href="#">Compare Source</a> environment.  Refer to <a href="#">Configuration Lab</a> for more information.	No	ENV-CODE = environment reference specified as a batch parameter on the batch run used to compare data.  DB-PROC-CODE = The <a href="#">Compare</a> DB process used to pull data from ENV-CODE.  <a href="#">MAX-ERRORS</a>	200/15
CL-COPDB	CIPYSYCB	The copy "CL_" DB processes is used to compare DB processes from the demonstration database. It is assumed that the demonstration database has been registered as a <a href="#">Compare Source</a> environment reference.  Refer to <a href="#">Configuration Lab</a> for more information.	No	ENV-CODE = environment reference representing a registered <a href="#">Compare Source</a> environment.  STATUS-ON-ADD = default root object status when root object action is <a href="#">Add</a> .  STATUS-ON-CHANGE = default root object status when root object action is <a href="#">Change</a> .  STATUS-ON-DELETE = default root object status when root object action is <a href="#">Delete</a> .  <a href="#">MAX-ERRORS</a>	200/15

## Archive and Purge Processes

The following table lists system background processes used in conjunction with the [Archive Engine](#). Note that the DB process that represents an archive or purge procedure defines the batch control used for the first step of a purge or archive. Although, the [CIPYCPRB](#) program (used universally for the first step of any archive or purge procedure) is delivered with the system, there are no system batch controls that reference it. It is included in this list for clarity.

For information on how to copy sample [Archive](#) and [Purge](#) DB processes from the demonstration database, refer to [How To Copy Samples From The Demonstration Database](#).

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
Special - Each archive or purge procedure defines its first step's batch control.	CIPYCPRB	The first step of any archive or purge process is to create primary archive roots objects.  Note that for <i>Purge</i> DB processes, the environment reference batch parameter is optional.  Refer to <a href="#">Archive Engine</a> for more information.	Yes	ENV-CODE = environment reference of the target archive environment.  TEST-MODE (Y/N) = If Y is specified, the program will write out information about primary roots to a log file.	200/15
AR-CRCHR	CIPYCRCB	The second step of any archive or purge process is to create child archive roots objects.  Note that for <i>Purge</i> DB processes, the environment reference batch parameter is optional.  Refer to <a href="#">Archive Engine</a> for more information.	Yes	ENV-CODE = environment reference of the target archive environment.  DB-PROC-CODE = The archive or purge DB process that specifies the batch control used to create primary archive root objects (step 1 of any archive or purge).  <a href="#">MAX-ERRORS</a>	200/15
AR-PRFK	CIPYRFB	The third step of any archive or purge process is to check recursive foreign key relationships.  Note that for <i>Purge</i> DB processes, the environment reference batch parameter is optional.  Refer to <a href="#">Archive Engine</a> for more information.	Yes	ENV-CODE = environment reference of the target archive environment.  DB-PROC-CODE = The archive or purge DB process that specifies the batch control used to create primary archive root objects (step 1 of any archive or purge).  <a href="#">MAX-ERRORS</a>	
AR-DCDT	CIPYPARB	The fourth step of any archive or purge process is to move data to the target archive environment (in the case of an archive), or simply delete it (in the case of purge).  Note that for <i>Purge</i> DB processes, the environment	No	ENV-CODE = environment reference of the target archive environment.	200/15

Batch Control ID	Program Name	Description	Multiple Threads	Extra Parameters	Records Between Commits Minutes Between Cursor Re Initiation
		reference batch parameter is optional. Refer to <a href="#">Archive Engine</a> for more information.		DB-PROC-CODE = The archive or purge DB process that specifies the batch control used to create primary archive root objects (step 1 of any archive or purge). <a href="#">MAX-ERRORS</a>	
AR-DCDTF	CIPYCARB	If desired, this background process may be run instead of <a href="#">AR-DCDT</a> (above). This process calls the <i>Archive Copy Processing</i> algorithm specified on the DB Process Instruction to copy the data to a flat file instead of an archive environment.  Note that for <i>Purge</i> DB processes, the environment reference batch parameter is optional. Refer to <a href="#">Archive Engine</a> for more information.	Yes	ENV-CODE = environment reference of the target archive environment.  DB-PROC-CODE = The archive or purge DB process that specifies the batch control used to create primary archive root objects (step 1 of any archive or purge).  MODE = C (Copy data only), D (Delete records only), or B (Both copy and delete). The default is B (Both).  <a href="#">MAX-ERRORS</a>	200/NA

## Column Descriptions

The following descriptions explain the parameters used in the above tables:

- **Batch Control ID.** As described earlier, every background process has an associated batch control record. This column contains the unique identifier of each process' batch control record.
- **Program Name.** This is the name of the program.
- **Description.** This column describes each background process.
- **Multiple Threads.** This column indicates if the background process uses the thread number and thread count to control parallel processing. Refer to [Parameters Supplied to Background Processes](#) for more information.

- **Extra Parameters.** This column indicates if the background process uses additional parameters (in addition to those described under [Parameters Supplied to Background Processes](#)).
- **Error Generates To Do.** This column indicates if the background process generates a To Do entry for object-specific errors as described in [Processing Errors](#).
- **Records Between Commits / Minutes Between Cursor Re-Initiation.** These values represent the maximum number of records between commits to the database and the number of minutes between cursor re-initiations. The process will issue a commit whenever the maximum records threshold has been exceeded. And, whenever a commit is issued, the process checks if the number of minutes between cursor initiation has been exceeded and if so, it will re-initiate the cursor. These values may be overridden when a specific background process is submitted. Refer to [Parameters Supplied to Background Processes](#) for more information.

## Batch Process Dependencies

---

The contents of this section illustrate the periodicity and dependencies between the various background processes described above.

### Contents

- [Batch Schedulers and Return Codes](#)
- [The Nightly Processes](#)
- [The Hourly Processes](#)
- [The XAI Processes](#)
- [The Letter Processes](#)
- [The Periodic Processes](#)
- [The ConfigLab Processes](#)
- [The Archive and Purge Processes](#)

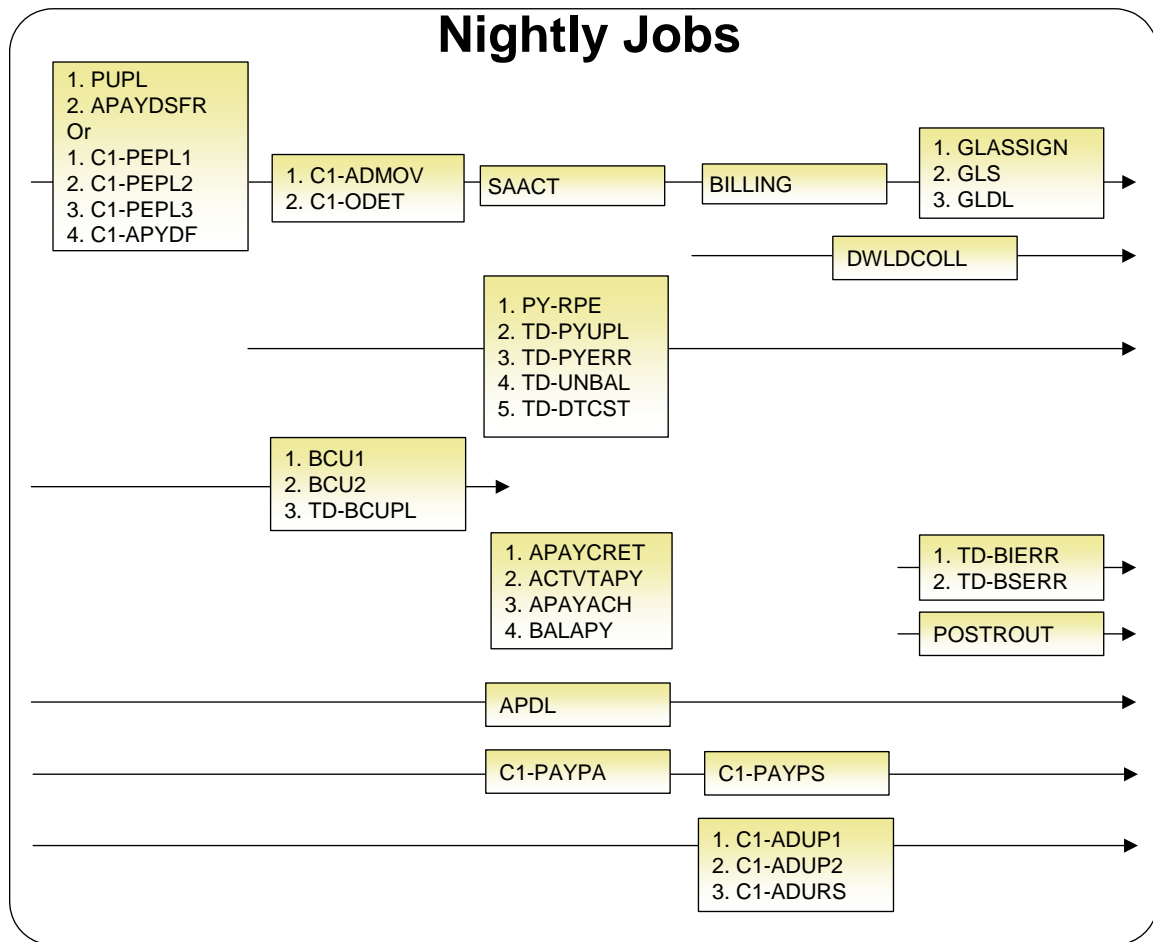
## Batch Schedulers and Return Codes

If you use a batch scheduler (e.g., Control-M, Tivoli) to control the execution of your batch processes, it will be interested in the possible values of each process's return code. The return code is a number that indicates if the process ended successfully. All product processes will return one of the following return code values:

- **0** (zero). A value of zero means the batch process ended normally.
- **2**. A value of 2 means the batch process detected a fatal error and aborted.

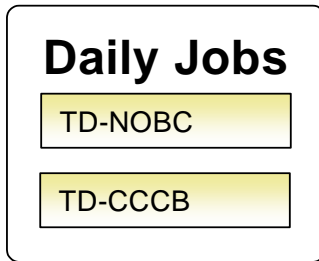
## The Nightly Processes

The following diagram illustrates the dependencies between the batch processes.



The mnemonics in the boxes refer to the individual batch processes described above. When a box contains multiple processes, these processes must be run sequentially. When multiple boxes exist on a timeline, all processes in an earlier box must execute before the subsequent box is executed. Those timelines that appear beneath the first job stream's timeline indicate when the timeline's respective processes can be executed in respect of the first job stream.

The following diagram illustrates the daily batch processes for which there are no dependencies.

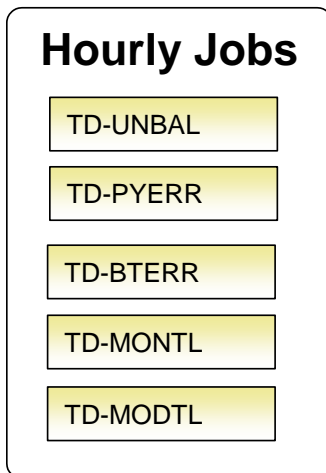


The mnemonics in the boxes refer to the individual batch processes described above.

**No dependencies exist.** As you can see, there are no dependencies between the boxes (meaning they may be run in parallel).

## The Hourly Processes

The following diagram illustrates the dependencies between the hourly batch processes.



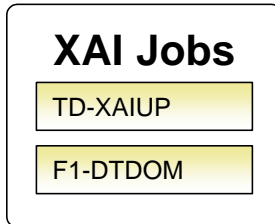
The mnemonics in the boxes refer to the individual batch processes described above. When a box contains multiple processes, these processes must be run sequentially.

**No dependencies exist.** As you can see, there are no dependencies between the boxes (meaning they may be run in parallel).



## The XAI Processes

The following diagram illustrates the dependencies between the XAI background processes. While these processes should be run at least once a day, you may want to consider running them more frequently (depending on how frequently you interface using XAI).

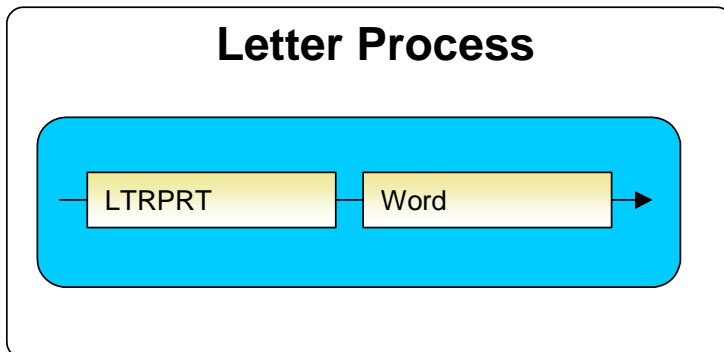


These processes create and/or clean up To Do entries for XAI upload staging or outbound messages in error. They are only applicable if your organization is using the XAI tool because only the XAI tool will mark one of these records in error.

## The Letter Processes

To extract information for your various letters, only one background process, **LTRPRT**, is required regardless of the different types of letters you have. This process simply calls an algorithm plugged-in on the respective letter template to construct its flat-file content.

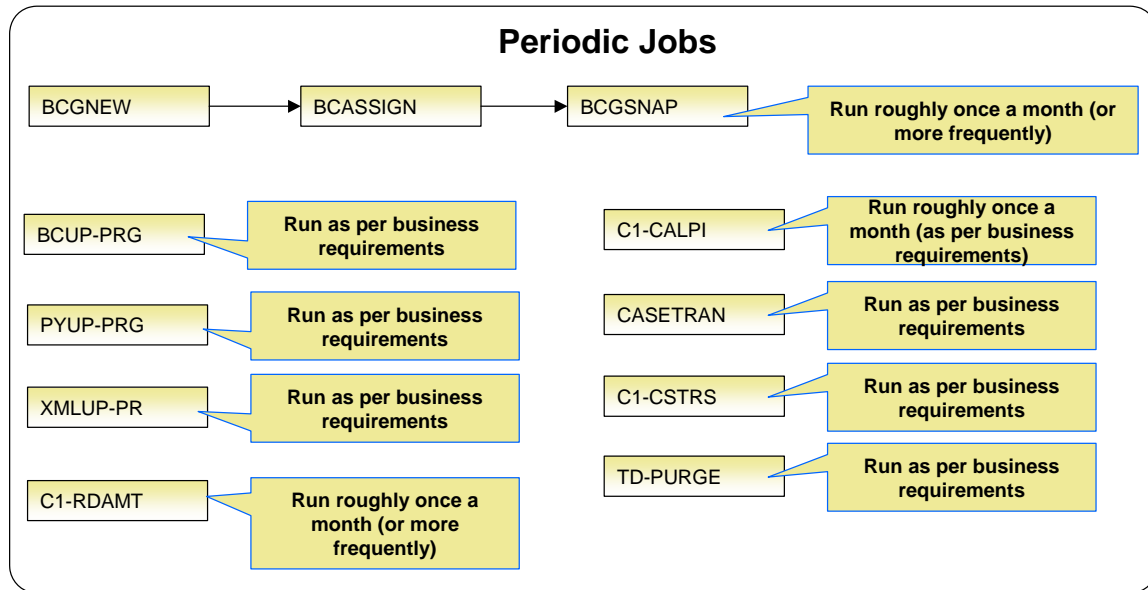
The following diagram illustrates the dependencies for the letter background process. While this process should be run at least on a daily basis, you may want to consider running it more frequently (depending on how frequently you produce letters).



The mnemonics in the boxes refer to the individual batch processes described above. When a box contains multiple processes, these processes must be run sequentially. When multiple boxes exist on a timeline, all processes in an earlier box must execute before the subsequent box is executed.

## The Periodic Processes

The following diagram illustrates the dependencies between the periodic background processes. While many of these processes should be run at least on a monthly basis, you may want to consider running them more frequently (depending on business requirements).

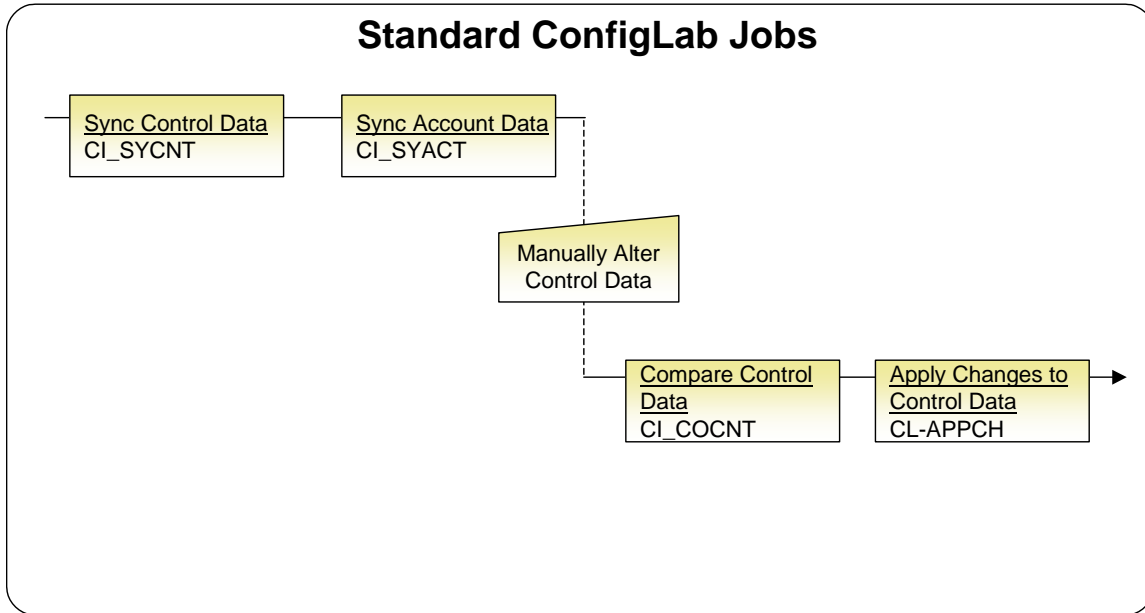


The mnemonics in the boxes refer to the individual batch processes described above.

**Few dependencies exist.** As you can see, there are few dependencies between the boxes (meaning they may be run in parallel).

## The ConfigLab Processes

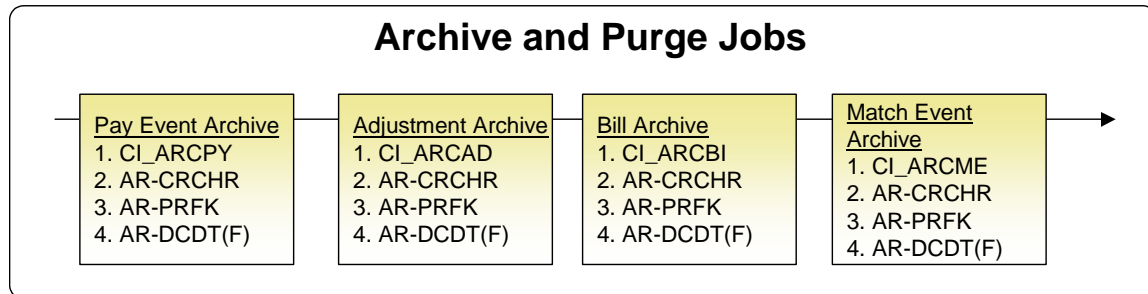
The following diagram illustrates the dependencies between the ConfigLab processes. The frequency of running ConfigLab processes is implementation specific. For more information on comparing data from an alternate environment, refer to [Configuration Lab](#).



**Compare Sources and Targets.** The [Configuration Lab](#) may be used with environments other than a **ConfigLab**. In cases where control and account data are pushed to a **Compare Target**, only the top two batch processes are executed. In cases where control data is pulled from a **Compare Source** environment, only the bottom two batch processes are executed.

## The Archive and Purge Processes

The following diagram illustrates the dependencies between the sample archive and purge processes. The frequency of running archive and purge processes are implementation specific. For more information on archive and purge processes, refer to [Archiving and Purging](#) and [Archive Engine](#).



**Steps 2, 3 and 4.** Note that steps 2 and 3 are the same for all of the sample archive and purge jobs. For step 4, you can select from two background processes: [AR-DCDT](#) moves data to a target archive environment (or purges the data) and [AR-DCDTF](#) calls an algorithm that copies the data to a flat file. If you want to use [AR-DCDTF](#) for other archive jobs, you must develop your own algorithms.

## How To Set Up A New Extract Processes

Several background processes delivered with the system are used to interface information out of the system. The topics in this section describe when and how to introduce an additional extract process.

### Setting Up Automatic Payment Extracts

You will need an automatic payment extract for every mechanism your company uses to route automatic payment requests to a financial institution / clearing house. For example:

- You will need an automatic payment extract to interface records to the Automated Clearing House (ACH) if you allow taxpayer to pay via credit card or direct debit from a checking account. The [APAYACH](#) process delivered with the system is intended to be used to handle this function.

If you need additional automatic payment extract processes, set up the following information:

- Add a new [batch control](#) record. Populate the fields as follows:
  - Batch Process.** Assign an easily recognizable unique ID for the automatic payment extract process.
  - Description.** Enter a description of the automatic payment extract process.
  - Accumulate All Instances.** Turn this switch on.
- Use [Auto Pay Route Type](#) to define the auto pay extract process to be used for each route type.

## The Big Picture of Sample & Submit

Sample and Submit refers to the ability to create Activity Requests. This is functionality that enables an implementer to design an ad-hoc batch process using the configuration tools.

Some examples of such processes are:

- Send a letter to customers that use credit cards for auto pay and the credit card expiration date is within 30 days of the current date.
- Stop auto pay for customers that use credit cards as the form of payment if the credit card has already expired. Notify the customer that their auto pay agreement has been terminated and that they need to call to reinstate.
- Select auto pay accounts that have more than X non-sufficient fund penalties, stop the auto pay agreement and notify the customer.

Note that the terms activity request and sample & submit request may be used interchangeably.

### Contents

- [Activity Type Defines Parameters](#)
- [Preview A Sample Prior To Submitting](#)
- [Credit Card Expiration Notice](#)
- [Exploring Activity Request Data Relationships](#)
- [Defining a New Activity Request](#)
- [Setting Up Activity Types](#)
- [Maintaining Sample & Submit Requests](#)

## Activity Type Defines Parameters

For each type of process that your implementation wants to implement, you must configure an activity type to capture the appropriate parameters needed by the activity request.

## Preview A Sample Prior To Submitting

To submit a new activity request, a user must select the appropriate activity type and enter the desired parameter values, if applicable.

After entering the parameters, the following actions are possible

- Click **Preview** to see a sample of records that satisfy the selection criteria for this request. This information is displayed in a separate map. In addition, the map displays the total number of records that will be processed when the request is submitted. From this map you can **Save** to submit the request, go **Back** to adjust the parameters or **Cancel** the request.
- Click **Cancel** to cancel the request.
- Click **Save** to skip the preview step and submit the request.

When an activity request is saved, the job is not immediately submitted for real time processing. The record is save in the status **Pending** and a monitor process for this record's business object is responsible for transitioning the record to **Complete**.

As long as the record is still **Pending**, it may be edited to adjust the parameters. The preview logic described above may be repeated when editing a record.

The actual work of the activity request, such as generating customer contact records to send letters to a set of customers, is performed when transitioning to **Complete** (using an enter processing algorithm for the business object).

## Credit Card Expiration Notice

The base product supplies a sample process to find customers that use credit cards for auto pay and the credit card expiration date is within x days of the current date.

To this functionality the following configuration tasks are needed:

- Define an appropriate [customer contact class](#) and [type](#) to use.
- Define appropriate activity request Cancellation Reasons. Cancellation reasons are defined using a customizable [lookup](#). The lookup field name is **C1\_AM\_CANCEL\_RSN\_FLG**.
- Define an activity type for the business object **C1-NotifyExpiringCreditCardTyp**. You may define default parameter values for the number of days for expiration and customer contact class and type.

## Exploring Activity Request Data Relationships

Use the following links to open the application viewer where you can explore the physical tables and data relationships behind the activity request functionality:

- Click [C1-ACM-ACTTY](#) to view the activity type maintenance object's tables.
- Click [C1-ACM-ACTRQ](#) to view the activity request maintenance object's tables.

## Defining a New Activity Request

To design a new ad-hoc batch job that users can submit via Sample and Submit, first create a new Activity Type business object. The base product BO for activity type **C1-NotifyExpiringCreditCardTyp** may be used as a sample.

The business object for the activity request includes the functionality for selecting the records to process, display a preview map for the user to review and to perform the actual processing. The base product BO for activity request **C1-NotifyExpiringCreditCardReq** may be used as a sample. The following points highlight the important configuration for this business object:

- Special BO options are available for activity request BOs to support the [Preview Sample](#) functionality.
  - Activity Request Preview Service Script. This script is responsible for retrieving the information displayed when a user asks for a preview of a sample of records.
  - Activity Request Preview Map. This is the map that is invoked to display the preview sample results.
- The enter algorithm plugged into the **Complete** state is responsible for selecting all the records that satisfy the criteria and processing the records accordingly.

## Setting Up Activity Types

Activity types define the parameters to capture when submitting an activity request via Sample and Submit. To set up an activity type, open **Admin Menu, Activity Type**.

The topics in this section describe the base-package zones that appear on the Activity Type portal.

### Contents

[Activity Type List](#)  
[Activity Type Actions](#)  
[Activity Type](#)

## Activity Type List

The Activity Type [List zone](#) lists every activity type. The following functions are available:

- Click a [broadcast](#) button to open other zones that contain more information about the adjacent activity type.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each activity type.
- State transition buttons are available to transition the activity type to an appropriate next state.

Click the **Add** link in the zone's title bar to add a new activity type.

## Activity Type Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions and appropriate state transition buttons are available.

## Activity Type

The Activity Type zone contains display-only information about an activity type. This zone appears when an activity type has been broadcast from the Activity Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

# Maintaining Sample & Submit Requests

Use the Sample and Submit transaction to view and maintain pending or historic activity requests. Navigate using **Main Menu, Batch, Sample & Submit Request**.

### Contents

[Sample & Submit Request Query](#)  
[Sample & Submit Request Portal](#)

## Sample & Submit Request Query

Use the [query portal](#) to search for an existing sample & submit request. Once a request is selected, you are brought to the maintenance portal to view and maintain the selected record.

## Sample & Submit Request Portal

This portal appears when a sample & submit request has been selected from the Sample & Submit Request Query portal.

The topics in this section describe the base-package zones that appear on this portal.

**Contents**

- [Sample & Submit](#)
- [Sample & Submit Actions](#)
- [Sample & Submit Log](#)

**Sample & Submit**

The Sample & Submit zone contains display-only information about an activity (sample & submit) request.

Please see the zone's help text for information about this zone's fields.

**Sample & Submit Actions**

This is a standard [actions zone](#).

Use the **Edit** button to modify the parameters. Refer to [Preview A Sample Prior to Submitting](#) for more information.

If the activity request is in a state that has valid next states, buttons to transition to each appropriate next state are displayed.

**Sample & Submit Log**

This is a standard [log zone](#).



# Defining Pay Plan Options

Tax authorities use pay plans as a collections tool. When a taxpayer agrees to be on a payment plan, it is an acknowledgement of his/her tax debt. Pay plans are typically used to satisfy obligations in which the taxpayer cannot satisfy the total obligation with one lump payment.

The pay plan's terms are typically negotiated between the tax authority and taxpayer. The payment amount and specific date intervals of the pay plan include such factors as total obligation amount and the taxpayer's ability to pay. Payments received are used to satisfy tax, penalty, and interest.

## Contents

[The Big Picture of Pay Plans](#)

[Setting Up Pay Plan Options](#)

## The Big Picture of Pay Plans

---

A pay plan is a special type of payment plan that encompasses two major elements:

- A set of scheduled payments
- The business rules used to recommend the payment schedule

Pay plans are managed via an obligation whose obligation type has a special role of **Pay Plan**. This type of obligations must have one or more recommendation rules, and are allowed to have payment schedules.

A pay plan's status is just like any other obligation's status. An active pay plan implicitly has a "kept" status (i.e. all scheduled payments have been made). A stopped plan implies that the pay plan has been completed, i.e. paid off, or stopped due to non-payment. It is important to note that the scheduled payments have their own status.

A list of the obligations covered by a pay plan is maintained with the pay plan. This list is used at payment distribution to determine which obligation to pay. An obligation may be covered by a pay plan when its obligation type is flagged as **Eligible for Pay Plan**.

## Contents

[Creating Pay Plans](#)

[Activating Pay Plans](#)

[Breaking Pay Plans](#)

[Stopping Pay Plans](#)

[Canceling Pay Plans](#)

[Distributing Payments for Pay Plans](#)

[Processing Scheduled Payments](#)

[Automatic Payment and Pay Plans](#)

[Alerts For Pay Plans](#)

## Creating Pay Plans

A payment plan can be created as a result of the following scenarios:

- A collections case is created as a result of overdue debt. One of the available actions for the user responsible for the collections case is setting up a pay plan. Refer to [The Big Picture of Collection Cases](#) for details on how a collection case can create a pay plan.

- The taxpayer may also voluntarily seek to enter a payment plan. Refer to [Maintaining Pay Plans](#) for details on how to create a pay plan manually.

## Activating Pay Plans

A pay plan is created in a **pending start** status. For the pay plan to become effective, it needs to be activated. If the pay plan does not require any special approval, the user can activate the pay plan once they have completed setting up the payment schedule. This can be done by navigating to the obligation page, and using the **activate** button.

If the pay plan requires approval, a different process might be implemented depending on the tax authority's rules.

When a pay plan is activated, you can perform special processing using an algorithm plugged in on the obligation Activation system event on the pay plan obligation type. The special processing can be developed to do anything that you would like, for example you could:

- Create a customer contact that with an appropriate letter template can generate a letter to inform the taxpayer of their payment amount and payment schedule.
- Initiate the creation of a payment coupon book for a taxpayer.
- In some cases, the taxpayer may be charged a setup fee for the pay plan. See the base algorithm ([C1-OT-LEVFEED](#)) for an example.

The system comes supplied with a sample algorithm type ([SAAT-CC](#)) that simply creates a customer contact to indicate that the pay plan is activated.

## Breaking Pay Plans

If the taxpayer does not meet their terms of the pay plan, the pay plan is considered broken. A pay plan can be broken in the one of the following ways:

- The user has used the **Break** action on the pay plan page.
- An automated process, such as monitor algorithm has determined that the taxpayer has broken the terms of the pay plan.

When the pay plan is broken, the pay plan is transitioned to **pending stop** status.

A base package break pay plan algorithm type ([C1-PAYP-BRK](#)) can be used to create a characteristic value to indicate if a pay plan is manually stopped by a user.

In addition, the algorithm type ([C1-OT-TPPRVW](#)) will create an account monitor review trigger if the pay plan is not associated with a collections case.

You can plug in an algorithm (of type [SAIS-ST](#)) on the obligation Stop Initiation system event on the pay plan obligation type to automatically stop the obligation (i.e. transition it to **stopped** status).

To finalize a pending stop obligation, the system calls the stop obligation algorithm plugged-in on the obligation Stop system event on the obligation type.

## Stopping Pay Plans

If the taxpayer pays off the outstanding debt, the pay plan is considered stopped. A pay plan can be stopped in one of the following ways:

- The collection case that created the pay plan was closed as a result of the taxpayer paying off the debt, and updates the pay plan to **pending stop**.
- An automated process, such as monitor algorithm has determined that the taxpayer has paid off the debt, and updates the pay plan status to **pending stop**.

As indicated above, an algorithm can be plugged it to transition the pay plan immediately from **pending stop** to **stopped**.

## Canceling Pay Plans

A pay plan may need to be cancelled under certain circumstances that are initiated either by the tax authority or the taxpayer. For example:

- A user may need to cancel an erroneous pay plan after it has already been activated.
- A tax authority may cancel pay plans (en masse) in the event of natural disasters.
- When a taxpayer indicates additional hardship, the procedure may involve canceling the existing pay plan and renegotiating a new pay plan.

Canceling a pay plan causes the pay plan status to change to **Cancelled**. Any specific actions that need to take place can be plugged in on obligation Cancel system event. The base package algorithm type ([C1-OT-CRCC](#)) can be used to create a customer contact when the pay plan is cancelled. Alternatively, ([SACA-CRTODO](#)) can be used to create a to do entry when the pay plan is cancelled.

## Distributing Payments for Pay Plans

For pay plans, payments are distributed directly to the covered obligations using payment distribution rules. A tax authority's business processes will dictate the distribution of payments to the corresponding obligations. For example:

- A tax authority wants payments to be applied to the oldest obligation first. It then moves to the next oldest obligation until all obligations are satisfied. The oldest obligation is given highest priority.
- A tax authority wants payments to be applied to obligations attached to an active collections case. The obligations linked to a collections case are assigned highest priority.
- The base-package provides an algorithm that will distribute payments to obligations covered by a pay plan based on age of debt, and priority indicated on the obligation type ([C1-DR-PAYPP](#)).

Refer to [Payment Event Distribution Rules](#) for details.

## Processing Scheduled Payments

In addition to the pay plan having a status, the scheduled payments each have their own status. A scheduled payment is created in a **pending** status when the pay plan is activated.

Each scheduled payment is reviewed by the base base-package batch **C1-PAYPS** on the payment due date. The batch will set the scheduled payment status to **processed** after executing the **Process Pay Plan Scheduled Payment** plug-in defined on the obligation type.

This plug-in can contain specific business logic that need to be executed when a scheduled payment is due. Based off the rules that will be configured by the tax authority, a multitude of actions can occur. They range from giving the taxpayer a few extra grace days to make the payment to canceling the current payment plan and forwarding the obligations to a collections authority. A tax authority's business rules as well as each taxpayer's personal history and circumstances should be taken into account when determining what actions to take next.

The base-package provides an algorithm ([C1-CC-CHKSP](#)) that will create a to do if a scheduled payment has not been paid. It will also detect when actual payments have fulfilled the pay plan in advance.

If your implementation's rules require additional status values (e.g. kept, late, partial), you will need to do the following:

- Create your own copy of [C1-PAYPS](#).
- Add custom values to lookup [NB\\_SCHED\\_PAY\\_STATUS\\_FLG](#).

## Automatic Payment and Pay Plans

If a taxpayer wants to pay their pay plan scheduled payments automatically, the account must be set up for automatic payment. In addition, the pay plan must indicate that automatic payment is being used.

Refer to [How To Set Up Automatic Payment For A Pay Plan](#) for more information.

When this is done, a background process referred to as [C1-PAYPA](#) creates automatic payments on the scheduled payment date by calling the automatic payment creation algorithm plugged in on the installation record.

The base-package provides two auto pay creation algorithms that support pay plans scheduled payments:

- Use algorithm type [APAY-CREATE](#) if your implementation distributes payments using the standard account type payment distribution.
- Use algorithm type [C1-APAY-CRDR](#) if your implementation distributes payments using distribution rules.

## Alerts For Pay Plans

The system provides [alerts](#) to highlight the existence of pay plans. These alerts are important to assist the tax authority's users:

- An alert is displayed if the account has a pay plan that is not stopped (e.g. ***pending start***, ***active*** or ***pending stop***).
- For taxpayers who are permanently forbidden from having a pay plan, the user should put a permanent [alert on the account](#).
- Use an algorithm to highlight cancelled pay plans with an entry in the alert zone. The algorithm type to do this is not provided. Use [C1-STOP-SA](#) as an example of how to create this type of algorithm.

- Configuring a specific [customer contact type](#) with the alert algorithm type ([CC BY TYPCL](#)) can cause alerts to be displayed whenever a user adds a [customer contact](#) of this type. This method can be used to highlight special conditions relating to the pay plan.

For more information about introducing alert conditions on Control Central, refer to [Installation Options - Algorithms](#).

## Setting Up Pay Plan Options

The topics in this section describe functionality that you must consider when designing pay plans.

### Contents

[Designing Recommendation Rules](#)  
[Setting Up The System To Enable Pay Plans](#)

## Designing Recommendation Rules

This section describes topics related to designing your recommendation rules.

### Contents

[What is a Pay Plan Recommendation Rule](#)  
[Examples of Recommendation Rules](#)  
[Generate Payment Schedule Algorithm Types](#)  
[Penalty and Interest Considerations](#)

## What is a Pay Plan Recommendation Rule

Users ask the system to recommend the scheduled payments for a pay plan. In general, this recommendation process must establish:

- The amount to be paid
- The dates on which the payments are due

A recommendation rule comprises of three elements:

- Payment schedule algorithms
- An algorithm to calculate an average daily amount. This is optional. If this plug-in must be used, it should be invoked from your payment schedule algorithms
- A collection of default parameter values for the payment schedule algorithm type

The default parameter values for the payment schedule algorithm type may change over time, so the collection contains an effective date. If default values are changed, these changes do not affect pay plans already in effect. Existing pay plans keep the parameter values that were used when the pay plan was started.

A user may override the default parameter values for the payment schedule algorithm type to customize the schedule if an override is allowed for a parameter. Additionally, a user may edit the payment schedule details at any time (provided the payment has not yet been processed).

**Note.** Normally parameter values for an algorithm type are kept with the algorithm. Because the parameters may vary for each pay plan, while the same algorithm logic is used, the parameter values are kept with the pay plan.

## Examples of Recommendation Rules

The following are two common methods of calculating a schedule of payments:

- Taxpayer knows how much and when they can pay. The recommendation rule will calculate the schedule of payment dates given the fixed payment amount, frequency, and the first payment date. For example: monthly payments of \$1,000 beginning on August 1, 2007.
- Taxpayer knows when they would like to pay the debt by, and how often they can pay. The recommendation rule will calculate the schedule of payment dates and payment amounts given the first payment date, the last payment date and the frequency. For example: monthly payments to be made starting on August 1, 2007 and ending on February 1, 2008.

## Generate Payment Schedule Algorithm Types

The core of the pay plan recommendation rule is the Generate Payment Schedule plug-in. The base-package provides two methods for generating a payment schedule: a fixed amount schedule or a fixed duration schedule. Both methods incorporate forecasted penalty and interest charges into the calculations of scheduled payment amount. For more details on these plug-ins see the following:

- A fixed amount payment schedule [C1-PP-FIXAMT](#).
- A fixed duration payment schedule [C1-PP-FIXDUR](#).

## Penalty and Interest Considerations

If [penalty and interest](#) (P&I) rules are applicable to the obligations covered by a pay plan, the payment schedule that is generated by the recommendation rule's generate payment schedule algorithm should factor expected penalty and interest calculations.

The penalty and interest plug-in spot may be called in [forecast](#) mode to forecast the charges to a certain date in the future. The plug-in spot also supports receiving financial transactions supplied by a calling service allowing for "what if" scenarios to be supplied. For example, when forecasting P&I to the future, the generate payment schedule algorithm can include proposed scheduled payments along with existing financial transaction to the P&I algorithm to get a more accurate estimate of the future P&I charges.

## Setting Up The System To Enable Pay Plans

The above topics provided background information about how pay plans are supported in the system. The topics in this section describe how to set up the system to enable pay plan functionality.

### Contents

- Characteristic Type
- Customer Contact Class And Types
- Algorithms
- Defining a Pay Plan Recommendation Rule
- Obligation Types

## Pay Plan Background Processes

### Characteristic Type

If your implementation requires a characteristic added to the pay plan when the pay plan is broken, you will need to create a characteristic type that specifies **obligation** as its characteristic entity.

### Customer Contact Class And Types

If your implementation requires certain customer contacts and/or letters created when the pay plan is activated or stopped, you will need to create appropriate customer contact class and customer contact type(s).

If you want to send letters to your taxpayers when a contact of any of these types is created, you must create an appropriate [letter template](#) and attach it to the customer contact type.

### Algorithms

As described above, in [The Big Picture of Pay Plans](#), the base package provides a number of algorithm types for different system events in the pay plan's lifecycle. In order to use any one of these algorithm types, you will need to create an associated algorithm and ensure any required parameters are populated.

If your implementation created additional custom algorithm types, you will also need to define associated algorithms.

### Defining a Pay Plan Recommendation Rule

Recommendation rules are used to recommend scheduled payments for pay plans. To define recommendation rules, navigate to **Admin Menu, Pay Plan Recommendation Rule**.

#### Description of Page

Enter an easily recognizable **Recommendation Rule** code and **Description** for each recommendation rule.

If applicable, specify the **Average Daily Amount Algorithm** used to calculate the average daily amount for this recommendation rule.

Specify the **Payment Schedule Algorithm Type** used to create the recommended payment scheduled for pay plans that use this recommendation rule. The Payment Schedule Algorithm Type cannot be modified if a pay plan that is not stopped or cancelled is using this recommendation rule.

**Payment Schedule Parameters** enables you to define collections of default parameter values for the payment schedule algorithm type that are effective dated. For each collection:

- **Effective Date** defines the date on which the collection of parameter values becomes effective.
- **Pay Plan Rule Parameter Value** specifies the default value of each parameter supplied to the algorithm. Note that the [payment schedule algorithm type](#) controls the number and type of parameters.
- **Override Flag** indicates whether the user can override the default value for the parameter.

**Where Used**

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_NB\\_RULE](#).

**Obligation Types**

This section provides information about setting up the obligation types for the pay plan itself and the obligation types for covered obligations.

**Contents**

- [Obligation Type for Obligations Covered by Pay Plans](#)
- [Pay Plan Obligation Type](#)

**Obligation Type for Obligations Covered by Pay Plans**

Consider which types of obligations need to be covered by a pay plan. For each obligation type that should be allowed to be covered by a pay plan, ensure that the **Eligible for Pay Plan** flag (on the Billing page) to **Eligible**.

**Pay Plan Obligation Type**

In order to enable pay plan functionality, you will need to define an obligation type that is suitable for a pay plan. Depending on the business rules of the tax authority, you may need multiple pay plan types, which will require you to define multiple obligation types. For example you might want to create a pay plan type for individual taxpayers, and a different pay plan type for a business.

The following points provide guidelines for creating a pay plan obligation type:

**Contents**

- [Obligation Type - Main](#)
- [Obligation Type - Detail](#)
- [Obligation Type - Billing](#)
- [Obligation Type - Rate](#)
- [Obligation Type - Algorithms](#)
- [Obligation Type - Pay Plan Recommendation Rule](#)

**Obligation Type - Main**

**Tax Type** should either reference a specific tax type or a generic tax type (e.g. miscellaneous fees) depending on your authority's business rules.

**Revenue Class** should be set to **N/A**. (Revenue classes are not applicable because pay plans do not apply a rate and revenue classes are only relevant for obligation types that use a rate.)

The **Payment Segment Type** should reference the **Normal Payment**.

**Do Not Overpay** should be on. Payments are not distributed directly against the pay plan.

**Obligation Type - Detail**

**Special Role** is **Pay Plan**.

**Obligation Type - Billing**

**Eligible for Billing** flag should be off, as pay plans do not get billed.

Additionally, the **Characteristic Location Required** should not be checked for pay plans.



**Eligible for Pay Plan** should be set to **Ineligible**. This option only applies to the obligation types that can be covered by a pay plan.

#### Obligation Type - Rate

Pay plans do not use rates, so the **Rate Required** flag should be off.

#### Obligation Type - Algorithms

Plug in any algorithms defined above for the following system events:

- Break Pay Plan
- Initiate Stop
- Obligation Activation
- Obligation Cancel
- Obligation Stop
- Process Pay Plan Scheduled Payment

#### Obligation Type - Pay Plan Recommendation Rule

Add the recommendation rules defined above that are valid for obligations of this type. Also, indicate which recommendation rule should be used as the default.

### Pay Plan Background Processes

Ensure that the following background processes are scheduled:

- Pay Plan Scheduled Payment Processing (**C1-PAYPS**)
- Pay Plan Scheduled Payment Automatic Payment Create (**C1-PAYPA**)

# Defining Case Management Options

Case management functionality is a highly configurable tool your organization can use to manage many situations, including (but certainly not limited to) the following:

- a formal appeal,
- a bankruptcy,
- a collection case,
- refund voucher processing,
- a taxpayer's request for literature,
- ... (the list is only limited by your time and imagination)

Obviously the steps involved in the resolution of the above cases are very different. The topics in this section describe how to configure the system to manage your cases as per your organization's desires.

**Separate module.** Please note that the Case Management functionality is associated with a separate module. If the **Case Management** module is not applicable to your business you may turn it off. Refer to [Turn Off A Function Module](#) for more information.

Refer to [Case Management](#) for a description of how end-users use cases.

## Contents

[The Big Picture Of Cases](#)  
[Setting Up Case Management Options](#)

## The Big Picture Of Cases

The topics in this section provide background information about how to configure the system to support your case management requirements.

## Contents

[Case Type Controls Everything](#)  
[Scripts and Cases](#)  
[ToDo's and Cases](#)  
[Examples of Case Types](#)

## Case Type Controls Everything

Whenever a user creates a case, they must specify the type of case (e.g., formal appeal, audit, etc.). The case type controls how the case is handled.

Case types hold the business rules that control cases. Since these business rules can sometimes be quite complicated, setting up case types requires planning and foresight. The topics in this section describe the type of business rules that can be configured on your case types.

## Contents

- Person / Account / Location Applicability
- Contact Information Applicability
- Business Object Association
- Additional Information
- Access Rights
- Lifecycle
- Status-Specific Business Rules
- Responsible User Applicability

## Person / Account / Location Applicability

Some types of cases may be person-oriented, others may be location-oriented, and still others may be account-oriented. For example:

- Cases used to keep track of a literature request would reference the person who requested the literature.
- Cases used to keep track of the inspection of a property would reference the location being inspected.
- Cases used to keep track of a formal appeal would reference the account associated with the bill(s) being questioned.

When you set up a case type, you define if its cases must reference a person, account, and/or location. Note, any combination of these objects is permitted on a case.

## Contact Information Applicability

When a case is created as a result of contact from an outside party, you may want to keep track of how to contact its originator. For example, you may want to record the originator's email address or phone number. When you set up a case type, you define if contact information is required, optional or not allowed on its cases.

## Business Object Association

A case type may reference a [business object](#), which serves as a link between cases of that type and the options that are associated with the business object.

## Additional Information

Some of your cases may require additional information (in the form of [characteristics](#)). For example, a formal appeal may require at least one bill. When you set up a case type, you can define the additional fields that are required. In addition, you can define default values for these fields.

The case functionality also allows you to require characteristics when a case enters a given state. Refer to [Required Fields Before A Case Enters A State](#) for the details.

**Requiring supporting documents.** Because any [type of characteristic](#) can be referenced on a case, you can require references to supporting documents by requiring a **file location** characteristic.

## Access Rights

You can take advantage of the system's [security](#) to restrict cases of a given type to certain users. For example, you can restrict bankruptcy cases to specific user groups.

The following points describe how to implement this type of security:

- Create an [application service](#) for each type of case you need to secure
- Define the access modes **Add**, **Inquire** and **Change** for each application service
- Define the applicable application service on each case type
- Link the appropriate [user groups](#) to each application service
  - For user groups that are allowed to add cases of a given type, define **Add** as a valid access mode.
  - For user groups that are allowed to view cases of a given type, define **Inquire** as a valid access mode
  - For user groups that are allowed to change cases of a given type, define **Change** as a valid access mode

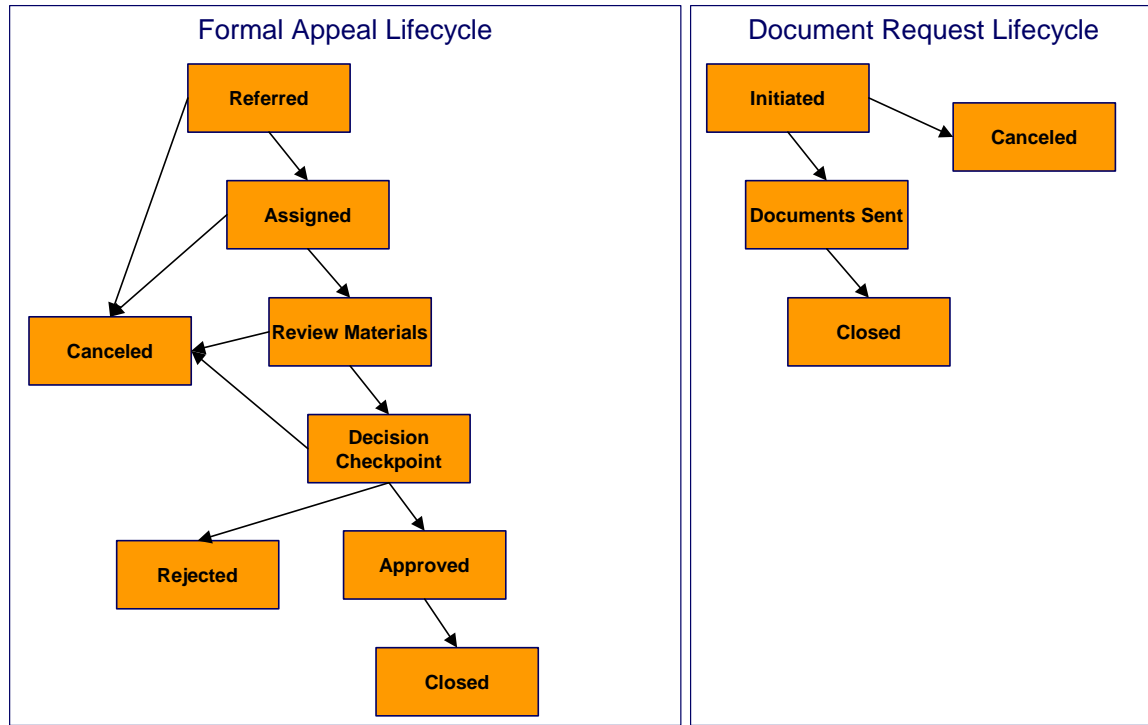
If you restrict access to a case type's cases, you can further restrict which users can work on cases given the status of the case. Refer to [Which Users Can Transition A Case](#) for more information.

**Restricting access to cases is optional.** If you don't specify an application service on a case type, all users (who have access to the case transaction) may access its cases.

## Lifecycle

Many objects in the system have predefined lifecycles whose rules are governed by the base-package and cannot be changed. For example, an obligation starts out in the **Pending Start** state and eventually becomes **Closed** when it's been final billed (and paid). You can't change the system to allow an obligation to start its life in the **Closed** state.

The lifecycle of cases is not governed by the base package. Rather, you define the lifecycle of your cases when you set up their case types. Examine the following diagrams; the one on the left shows the potential lifecycle of a case that manages a formal appeal, the one on the right shows the potential lifecycle of a case that manages a taxpayer's documentation request.



Potential Lifecycles Of Two Types Of Cases

**Just examples.** The above lifecycles are just examples. When you set up your case types, you must define the valid states for your case type.

The topics that follow describe important concepts that are illustrated in the above diagrams.

## Contents

- Valid States versus State Transition Rules
- Transitory States
- One Initial State and Multiple Final States
- Allowing A Case To Be Reopened
- Make Sure To Have A Canceled State
- Buttons Allow A User To Transition A Case From Status To Status
- State Transitions Are Audited
- Reports and Analytics Highlight Productivity

## Valid States versus State Transition Rules

The orange boxes in the above diagram show the potential valid states a given case can have. The lines between the boxes indicate the state transition rules. These rules govern the states a case can move to while it's in a given state. For example, the above diagram indicates a formal appeal that's in the **Referred** state can be either **Canceled** or moved into the **Assigned** state.

When you set up a case type, you define both its valid states and the state transition rules.

### Transitory States

You can define a state in a case type as **Transitory** if you do not wish the case to exist in a particular state. For example, let's assume that an algorithm is associated with the **Decision Checkpoint** state (Enter Processing) that would automatically determine the next state for the case (i.e. **Approved** or **Rejected**) and also contains logic to transition the case accordingly. In this scenario, you may not ever want the case to exist in the **Decision Checkpoint** state, so that a user won't ever see a formal appeal in that state. If the other states were marked as **non-transitory**, and an error were to occur during the transition from **Decision Checkpoint** to **Approved**, the case would roll back any changes to data made in the **Approved** (Enter Processing) state along with the changes made in the **Decision Checkpoint** state, and would end up in the **Review Materials** state - the last non-transitory state prior to **Decision Checkpoint**.

### One Initial State and Multiple Final States

When you set up a case type's states, you must pick one as the initial state. The initial state is the state assigned to new cases of a given type. For example, formal appeal cases have an initial state of **Referred**, whereas document request cases have an initial state of **Initiated**.

You must also define which statuses are considered to be "final". When a case enters a "final" state, it is complete and no further action is necessary. You might want to think of the "final" states as the potential outcomes of a case. For example, a formal appeal has potential outcomes of **Closed**, **Rejected**, and **Canceled**.

The "final" states are used by the system to differentiate between open and closed cases. For example, an alert highlights when the person / account / location in context has open cases (this alert only exists if you've plugged-in the appropriate installation [alert](#)).

### Allowing A Case To Be Reopened

You can set up your state transition rules to allow a case to be reopened (i.e., to be moved from a final state to a non-final state). Neither of the above examples allows this, but it is possible if you configure a case type accordingly.

### Make Sure To Have A Canceled State

The system does not allow you to delete a case. Therefore, if you want to support logical deletion, you should have a status of **Canceled** early in a case type's lifecycle. Doing this allows a user to cancel (i.e., logically delete) a case.

**Cancel reason.** You might want to consider setting up your case types to require a cancel reason (in the form of a [predefined value characteristic](#)) when a user cancels a case. Refer to [Required Fields Before A Case Enters A State](#) for more information.

### Buttons Allow A User To Transition A Case From Status To Status

When a case is displayed on [Case - Main](#), a separate button is shown for each state into which the case can be transitioned. For example, a formal appeal case that is in the **Referred** state would show two buttons: **Assign** and **Cancel**. If the user presses the **Assign** button, the case is transitioned to the **Assigned** state. If the user presses the **Cancel** button, the case is moved to the **Canceled** state.

You may define the text displayed on the button differently for each state transition. This allows the action description to be varied according to the previous status. For example, the button to transition from **New** to **Active** may be labeled **Activate**, but the button to change from **Closed** to **Active** may be labeled **Reactivate**.

Refer to [Which Users Can Transition A Case](#) for instructions describing how to restrict users to specific actions.

### State Transitions Are Audited

The system maintains an audit trail whenever a case transitions from one state to another. This audit is shown in the case's [log](#).

### Reports and Analytics Highlight Productivity

When you set up a case type's lifecycle, keep in mind that several reports and analytics highlight how long it took cases to transition into a state. For example, you can use a report to see how long it took formal appeal to be completed (or initially worked on or ...). Refer to the [Reports](#) chapter for the details of case reports.

## Status-Specific Business Rules

As described in [Lifecycle](#), when you set up a case type, you define the possible states its cases can pass through. The following topics describe business rules that can be configured for each state.

### Contents

- [A Script That Helps A User Work Through A Case](#)
- [Required Fields Before A Case Enters A State](#)
- [Validation Before A Case Enters A State](#)
- [Additional Processing When Entering A State](#)
- [Validation Before A Case Exits A State](#)
- [Additional Processing When Exiting A State](#)
- [Automatic Transition Rules](#)
- [Script Launching Option](#)
- [Which Users Can Transition A Case Into A State](#)

### A Script That Helps A User Work Through A Case

You can define a [Business Process Assistant script](#) that helps a user work a case while it's in a given state. For example, when you set up the **Review Materials** state for the formal appeal case type, you can define a script. A user can then easily launch this script to help them work through a case in this state.

Please keep the following in mind when you're designing how to integrate BPA scripts with your cases:

- You can have a different script for each state. For example, you could develop a script to help a user work on a case while it's in the **Review Materials** state and a different script to help them work in a case while it's in the **Approved** state.
- Rather than make a user launch a script by pressing a hyperlink on the [case page](#), you can have the system automatically launch the script while the case is in a given state. Refer to [Script Launching Option](#) for more information.
- You can also have the system automatically launch a script when a user selects a ToDo entry. Refer to [Launching Scripts When ToDo Entries Are Selected](#) for more information.

Refer to [Scripts and Cases](#) for more information about how to streamline your case processing with scripts.

### Required Fields Before A Case Enters A State

You can define additional fields (i.e., characteristics) that are required before a case can enter a given state. For example,

- You can indicate a formal appeal must reference at least one bill before it enters the **Referred** state.
- You can indicate a case must reference a cancel reason before it enters the **Canceled** state.

You do this by indicating that [characteristics](#) (that were optional when the case was added) are required when a case enters a given state.

### Validation Before A Case Enters A State

You can define validation that executes before a case can enter a given state. For example, you can indicate the case must have been assigned a responsible user before it can enter the **Assigned** state. This validation logic is held in algorithms that are plugged in on the case type and therefore you can define any type of validation.

### Additional Processing When Entering A State

You can define additional processing that should happen when a case enters a given state. For example, you can have a [letter](#) created when a formal appeal case is **Rejected**. Similarly, you can have a [To Do entry](#) created when a formal appeal enters the **Assigned** state. This additional processing is held in algorithms that are plugged in on the case type and therefore you can define any type of additional processing.

You can also incorporate state transition logic within routines that are executed when a case enters a state, so that you do not need to rely upon CASETRAN to transition your cases. For example, when the state entry routines of the **Decision Checkpoint** status for a formal appeal are executed, they may be designed to transition the case into either the **Rejected** or **Approved** state without waiting based on fields linked to the case. Note that your Exit Validation and Exit Processing logic, if configured for the case state, will still be executed as part of the state transition. Auto-Transition logic for this state will be ignored during this transition.

### Validation Before A Case Exits A State

You can define validation that executes before a case can exit a given state. For example, you might want to check the account's balance is less than a given value before a collection case can exit a given state. This validation logic is held in algorithms that are plugged in on the case type and therefore you can define any type of validation.

### Additional Processing When Exiting A State

You can define additional processing that should happen when a case exits a given state. For example, you can have a [To Do entry](#) automatically completed when a formal appeal leaves the **Decision Checkpoint** state. This additional processing is held in algorithms that are plugged in on the case type and therefore you can define any type of additional processing.



### Automatic Transition Rules

You can define rules that automatically transition a case into a different state using auto transition rules. For example, if you have a case that sends a letter to a taxpayer, it can be configured to transition to the **Follow Up** state 1 week after the letter is sent. For a collection case, you can configure any state to automatically transition to **Canceled** if the taxpayer pays their debt in full. These rules are held in algorithms that are plugged in on the case type and therefore you can define any type of automatic transition rules.

Cases in a state with automatic transition rules are monitored by the [CASETRAN](#) background process. Each time this program runs, the respective automatic transition plug-in is called for each such case and it transitions the case if the condition applies.

When the user adds a new case or changes the state of a case manually the system attempts to auto-transition the case to subsequent statuses as necessary. If auto-transition rules apply to the new state (and to subsequent ones) they would be executed right away. In other words, you don't need to wait for the auto-transition background process to be executed. An indication that the case was auto-transitioned online is displayed right below the action buttons section.

**Auto-Transition Errors.** Online auto-transition is performed recursively committing each successful state transition to the database. It is performed up to 100 times or until an error is encountered during the process. If this happens, auto-transition stops at the last **non-transitory** state into which a successful transition had occurred. Two case log entries will be generated automatically - one containing the message that a transition error has occurred, and a second containing the actual error message. A To Do entry will also be generated automatically upon rollback. The type of this To Do entry will be taken from 1) the **Case Transition Exception To Do Type** option for the **Business Object** associated with the case type, and if this is not populated, 2) the **Exception To Do Type** indicated on the Case Options Feature Configuration. All of the above error handling is true for both batch and online processing of cases.

**Triggering Auto-Transition.** If you have a customized process that affects the state of a case and you want the case to be auto-transitioned right away, i.e. not wait for the next scheduled [CASETRAN](#) background process to execute, you can customize that process to trigger auto-transition for the specific case, or you can put the state transition logic into the routines that execute at state entry time.

### Script Launching Option

You can define whether the script associated with a given state is to be automatically launched while the case is in that state. The system supports the following options:

- Launch the script only if no script is currently active.
- Always launch the script unless this specific script is currently active.

**Warning!** With this option, if a script is currently open in the page's BPA script area then it will be automatically closed and the case script will open.

- Do not automatically launch the script.

You do this by plugging-in a **Script Launching** algorithm for the given state. If no such plug-in is provided the script is not automatically launched.

### Which Users Can Transition A Case Into A State

If you have [restricted access](#) to a case type, you can further restrict which user groups are allowed to transition a case into specific states. For example, you can control which user group can transition a formal appeal into the **Assigned** state. The following points describe how this is done:

- Define actions on the [application service](#) defined on the case type. You must define an action for each status that you need to secure.
- Define each status's corresponding action. Note, you only need to link a status to an action if it's secured. Any user with [access](#) to the case type can perform statuses that aren't linked to actions.
- Define the transition role for each status's valid next status. You can assign valid next statuses to be reachable via system (only), or system and user.
- Define which [user groups](#) have access to the actions (i.e., statuses). In addition, these user groups should have access to the **Change** action.

### Responsible User Applicability

Some of your cases may require a "responsible user". This is the user who has overall responsibility for the case. When you set up a case type, you define if a responsible user is required, optional or not allowed on its cases.

The following points describe how to set up the system if a responsible user is not required when a case is first created, but is later in its lifecycle:

- Indicate that a responsible user is optional on the case type
- Plug-in either an [exit validation](#) or [entry validation](#) algorithm on one of the case type's states to require a responsible user at some point in a case's lifecycle

**Address To Do entries to the responsible user.** If you use the [base-package algorithm](#) to create a To Do entry when a case enters a given state, you can indicate that the To Do entry should be addressed to the responsible user on the case.

## Scripts and Cases

There are three ways [Business Process Assistant scripts](#) can be used to manage cases:

- You can create a BPA script to help users create a case. For example, a script can help a user create a new formal appeal.

Using a script to create a case can save a user a lot of time (and training efforts). This is because the script can automatically populate many fields on the case based on answers to questions.

Refer to [Initiating Scripts](#) for a description of how end-users initiate scripts.

- You can create a script to help users work on a case when it's in a given state. Refer to [A Script That Helps A User Work A Case](#) for more information.

- You can [set up your case types to create ToDo entries](#) to notify users when cases exist that require their attention. Users can complete many of these ToDo entries without assistance. However, you can set up the system to automatically launch a script when a user selects a ToDo entry. For example, consider a ToDo entry that highlights a formal appeal that requires investigation. You can set up the system to execute a specific script when a user selects this ToDo entry. This script might guide the user through the investigation process (and help them update the case). Refer to [Executing A Script When A To Do Entry Is Selected](#) for more information.

## ToDo's and Cases

The topics in this section provide background information about how to facilitate case management with [To Do entries](#).

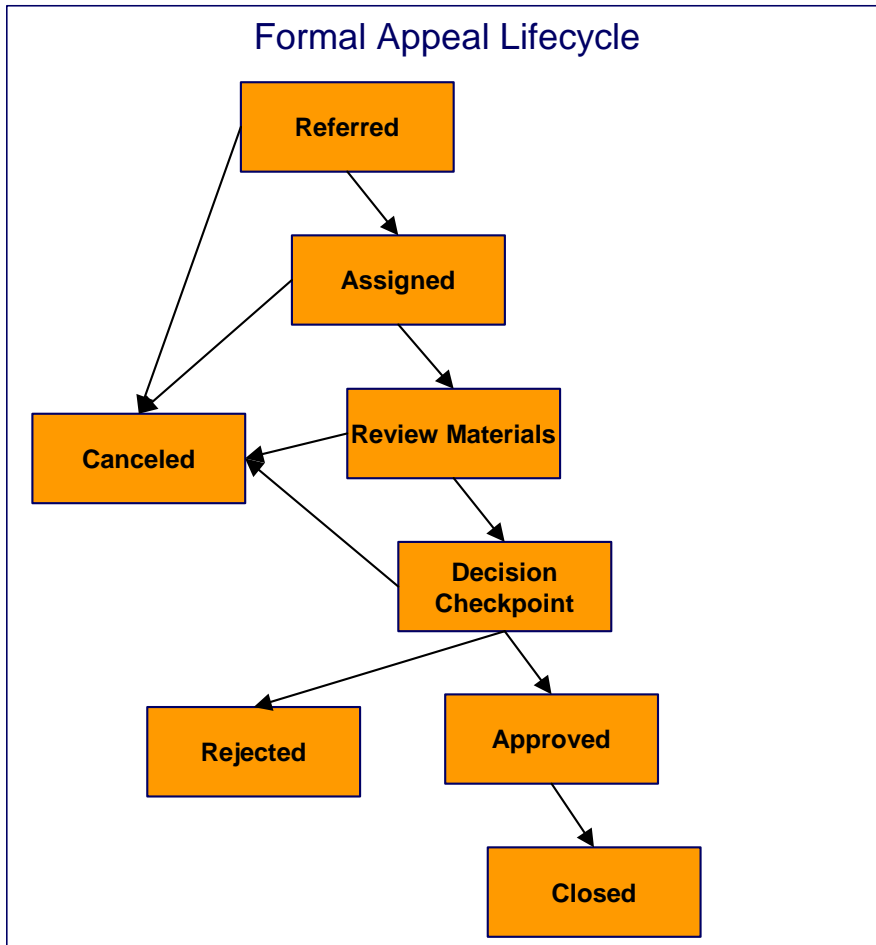
### Contents

- [Creating and Completing ToDo Entries](#)
- [Launching Scripts When ToDo Entries Are Selected](#)
- [All ToDo Entries Are Visible](#)

### Creating and Completing ToDo Entries

You can configure your case types to create and complete [To Do entries](#) when a case enters or exits a state.

Let's use the following lifecycle diagram to illustrate this point:



Let's assume the following:

- You want a ToDo entry created when a formal appeal is created and you want it completed when the case reaches the **Canceled**, **Rejected** or **Approved** states. This ToDo entry could be used by a supervisor to monitor the number of formal appeal being worked. Note, we refer to this as the "first" ToDo entry below.
- You want a different ToDo entry created when the case enters the **Assign** state and you want this entry automatically completed when the case leaves this state. Note, we refer to this as the "second" ToDo entry below.
- You want a different ToDo entry created when the case enters the **Decision Checkpoint** state and you want this entry automatically completed when the case leaves this state. Note, we refer to this as the "third" ToDo entry below.

To implement the above, you would set up the case type as follows:

- Plug-in an [entry processing](#) algorithm on the **Referred** status to create the first ToDo entry. Plug-in an [entry processing](#) algorithm on the **Canceled**, **Rejected** and **Completed** statuses to complete this entry.

- Plug-in an [entry processing](#) algorithm on the **Assigned** status to create the second ToDo entry. Plug-in an [exit processing](#) algorithm on the **Assigned** status to complete this entry. We elected to use an exit processing algorithm because we only have to plug it in on one status. If we'd used an entry processing algorithm, we would need to plug it in on the 2 statuses into which the **Assigned** status can transition.
- Plug-in an [entry processing](#) algorithm on the **Decision Checkpoint** status to create the third ToDo entry. Plug-in an [exit processing](#) algorithm on the **Decision Checkpoint** status to complete this entry.

## Launching Scripts When ToDo Entries Are Selected

You can set up your case types to create ToDo entries to notify users when cases exist that require their attention. Users can complete many of these ToDo entries without assistance. However, you can set up the system to automatically launch a script when a user selects a ToDo entry. For example, consider a ToDo entry that highlights a formal appeal that requires investigation. You can set up the system to execute a specific script when a user selects this type of ToDo entry. This script might guide the user through the investigation process. Refer to [Executing A Script When A ToDo Entry Is Selected](#) for more information.

## All ToDo Entries Are Visible

When a case is displayed on [Case Maintenance](#), the system summarizes the number of ToDo entries associated with the case (if you've [set up your ToDo types](#) appropriately).

## Examples of Case Types

The topics that follow provide an example of a case type related to a formal appeal. Use the information in this section to form an intuitive understanding of case types. After attaining this understanding, you'll be ready to design your own case types.

### Formal Appeal Case Type

Some organizations will set up a case to manage a formal appeal. The following diagram illustrates how such a case type might look:

### Case Type – Formal Appeal

**Person / Account / Premise Applicability**

Field	Applicability
Person	Optional
Account	Required
Location	Required

**Secured**

Yes, all actions

**Fields To Be Captured**

Field	Required / Optional	Default Value
Bill ID	Optional	‘ ‘
Cancel reason	Optional	‘ ‘
Reject reason	Optional	‘ ‘
Assessment change	Optional	‘ ‘

**Applicability Rules**

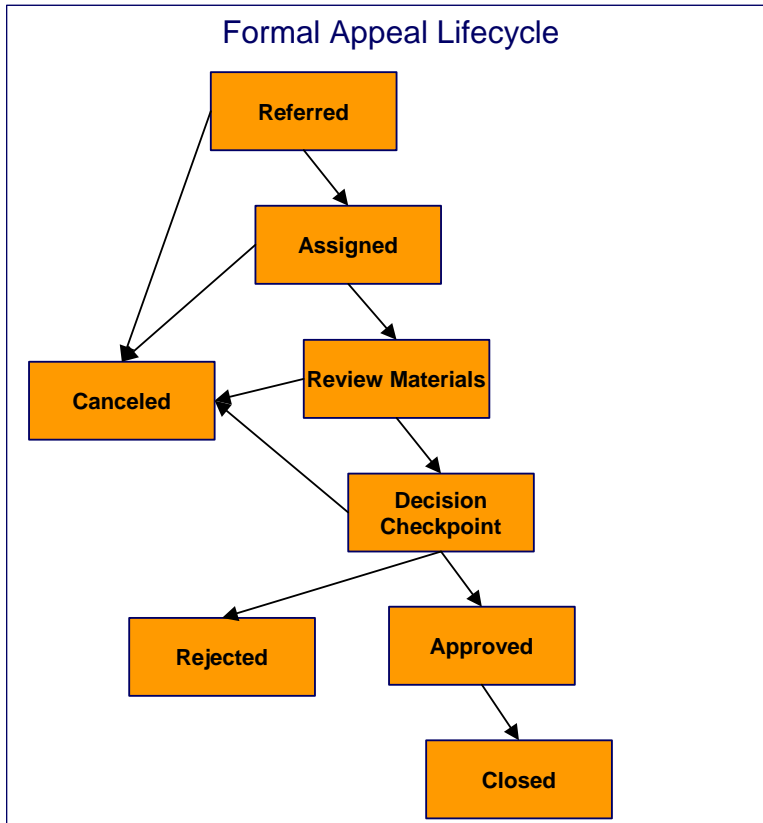
Field	Applicability
Responsible User	Optional
Contact Information	Optional

Note the following about the formal appeal case type:

- Notice that we set up the case type to require an account and a location, but person is optional. This is because multiple persons may own a property, whose assessed value is being appealed. Different tax types will need different required and optional parameters.
- We need to restrict formal appeal to specific user groups. This means we need to [set up a specific application service](#) for this case type (that has a separate "action" for each status).
- Cases of this type have 3 additional fields over their lifetime. Notice the following:
  - The **Bill ID** characteristic is set up to be optional. This is because the assessment may or may not have been billed yet. We've assumed that sometimes a formal appeal case can be created before the tax assessment has been billed.
  - Both the **Cancel Reason** and **Reject Reason** are optional. Later in this section, you'll see that we've configured the Canceled and Rejected statuses to require these fields, respectively.

- Cases of this type do not need a **Responsible User** when first created. Later in this section, you'll see how we've configured the **Assigned** status to require a **Responsible User** before a case can enter this state.
- Cases of this type do not need **Contact Information**. This was a subjective decision and depends on your organization's philosophy.

The topics that follow describe each of the statuses in a formal appeal [lifecycle](#). We have assumed the following state transition rules:

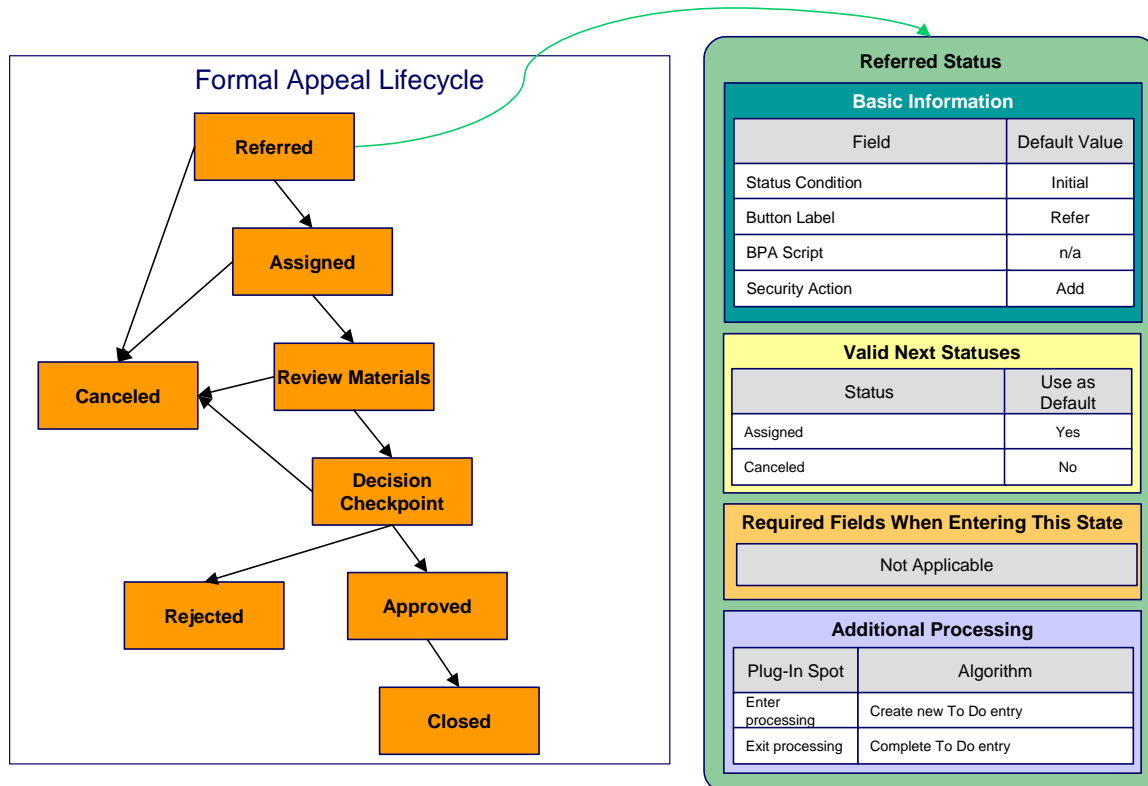


## Contents

[Referred Formal Appeal](#)  
[Assigned Formal Appeal](#)  
[Review Materials Formal Appeal](#)  
[Decision Checkpoint Formal Appeal](#)  
[Approved Formal Appeal](#)  
[Closed Formal Appeal](#)  
[Rejected Formal Appeal](#)  
[Canceled Formal Appeal](#)

## Referred Formal Appeal

The following is an example of the configuration of the **Referred** status for formal appeal cases.



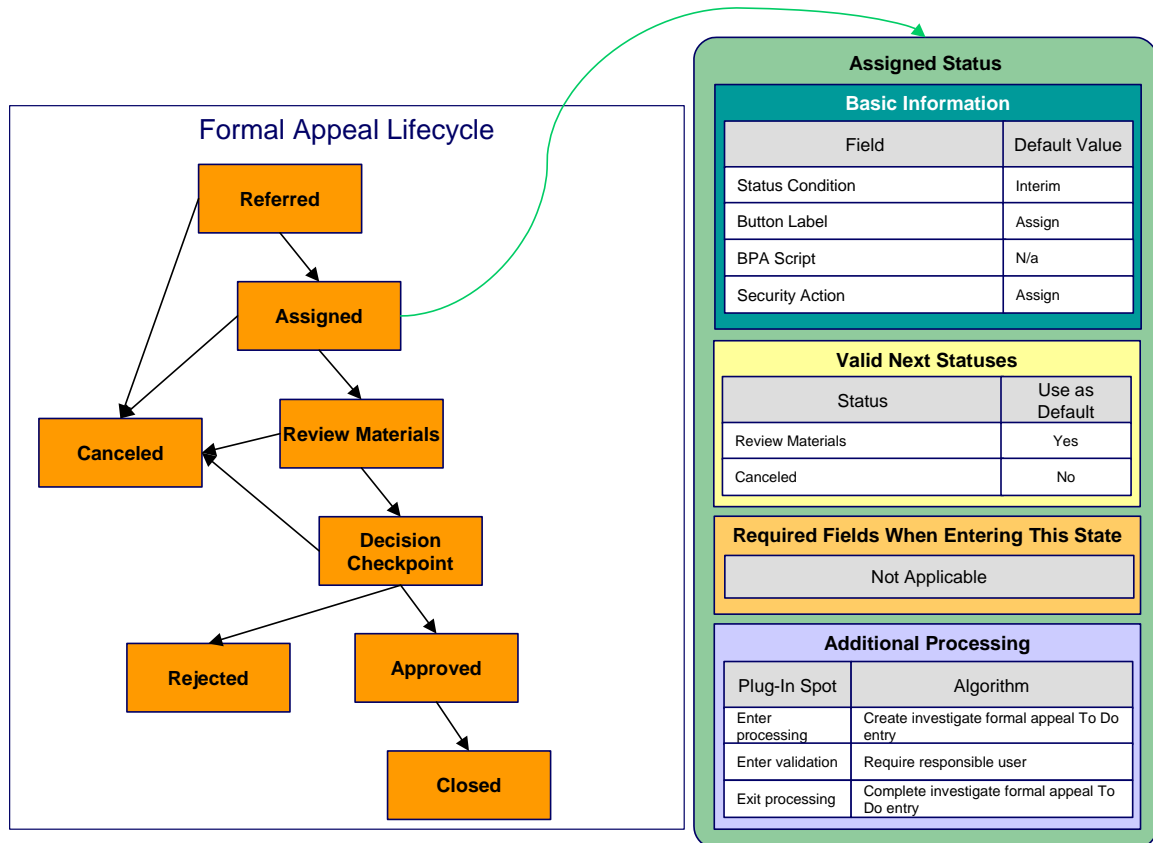
Note the following about this status:

- **Referred** is the initial state. The initial state is the state populated on new cases of a given type. Remember that only one status can be defined as the initial state.
- It has a button label even though it's the initial state. The above diagram doesn't allow a user to ever transition a case into this state and therefore there will never be a button with such a label. However, it's a required field just in case you change your business rules.
- We have decided not to use a BPA script to help a user work on a formal appeal when it's in this state (this is probably not the best decision as BPA scripts can be quite useful).
- We have associated the Add action with this status. This means that only users with rights to the add action for the application service defined on the case type can add cases of this type.
- Notice that Valid Next Statuses are what restricts a case in this state to be transitioned to the **Canceled** and **Assigned** states.
- Notice that the Additional Processing plug-ins create and complete a ToDo entry when a case enters and exits this state, respectively.

### Assigned Formal Appeal

The following is an example of the configuration of the **Assigned** status for formal appeal cases.



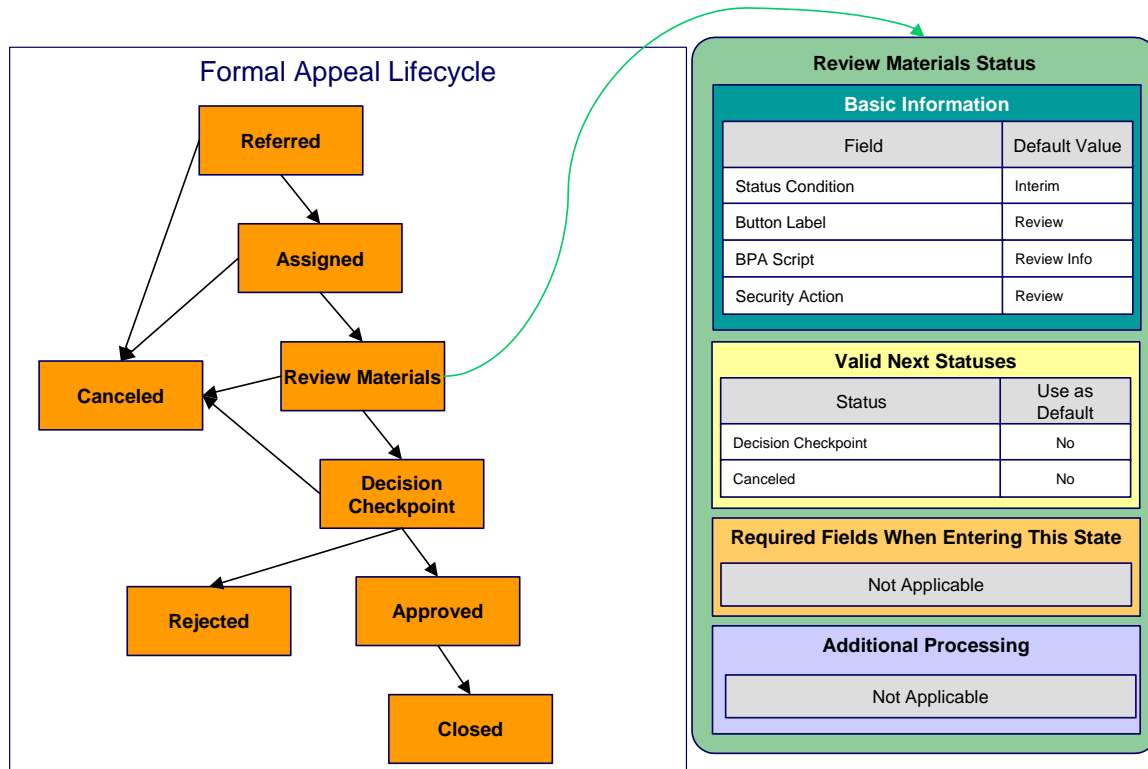


Note the following about this status:

- **Assigned** is an interim state (meaning that it's not an initial or final state).
- It has a button label of **Assign**. This is the label on the button that a user presses to move a case into this state. This button will only appear on cases that are in the **Referred** state as this is the only state that can transition into the **Assigned** state.
- We have decided not to specify a BPA script on this status. Rather, we're going to set up the ToDo type used to highlight cases in this state to automatically launch an appropriate BPA script when a user selects the ToDo entry.
- We have associated the **Start Investigation** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that Valid Next Statuses are what restricts a case in this state to be transitioned to the **Review Materials** and **Rejected** states.
- Notice that the Additional Processing plug-ins do the following:
  - Create and complete a ToDo entry when a case enters and exits this state, respectively
  - Require a responsible user before a case can enter this state

### Review Materials Formal Appeal

The following is an example of the configuration of the **Review Materials** status for formal appeal cases.

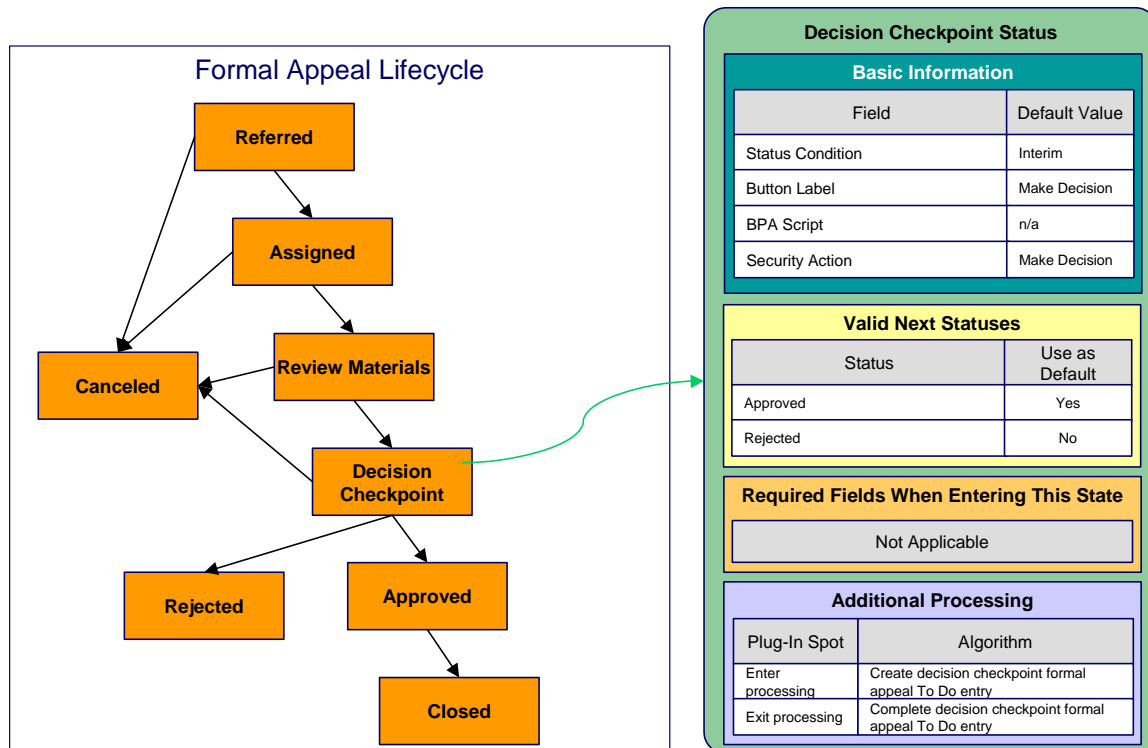


Note the following about this status:

- **Review Materials** is an interim state (meaning that it's not an initial or final state).
- It has a button label of **Review**. This is the label on the button that a user presses to move a case into this state. This button will only appear on cases that are in the **Assigned** state as this is the only state that can transition into the **Review Materials** state.
- We have decided to specify a BPA script on this status to help users find information about the property to help in reviewing the case.
- We have associated the **Review** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that Valid Next Statuses are what restricts a case in this state to be transitioned to the **Decision Checkpoint** and **Canceled** states.

### Decision Checkpoint Formal Appeal

The following is an example of the configuration of the **Decision Checkpoint** status for formal appeal cases.

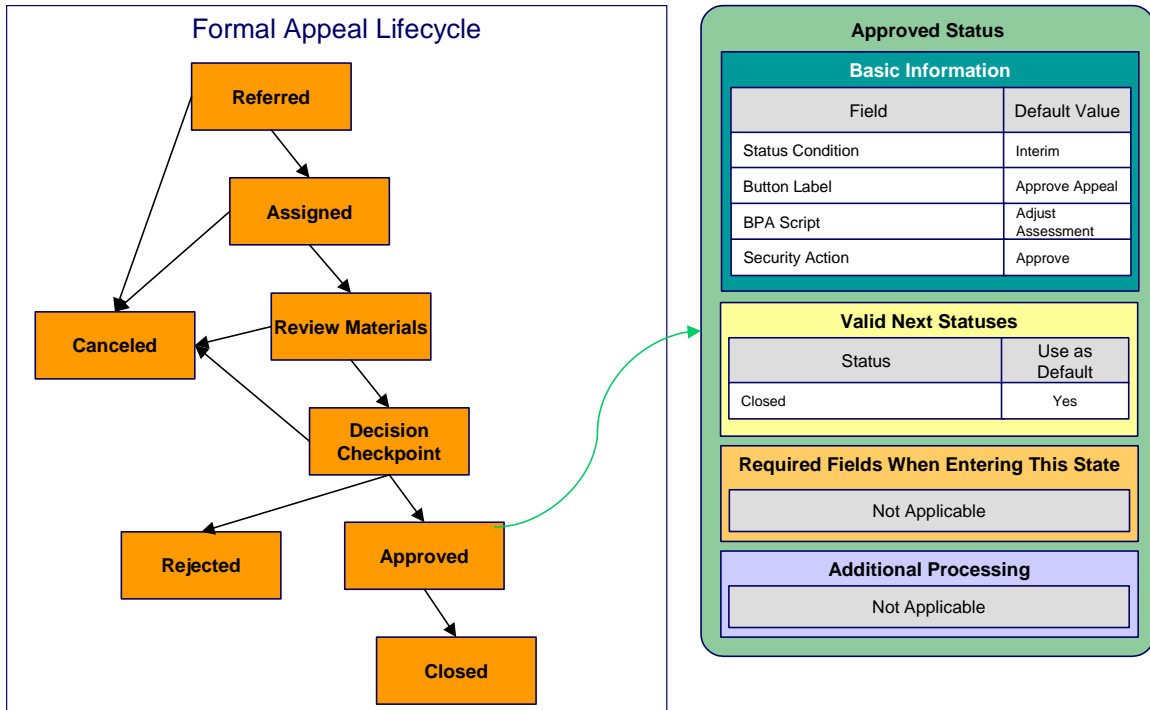


Note the following about this status:

- **Decision Checkpoint** is an interim state (meaning that it's not an initial or final state).
- It has a button label of **Make Decision**. This is the label on the button that a user presses to move a case into this state. This button will only appear on cases that are in the **Review Materials** state as this is the only state that can transition into the **Decision Checkpoint** state.
- We have decided not to specify a BPA script on this status. Rather, we're going to set up the ToDo type used to highlight cases in this state to automatically launch an appropriate BPA script when a user selects the ToDo entry.
- We have associated the **Make Decision** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that Valid Next Statuses are what restricts a case in this state to be transitioned to the **Approved** and **Rejected** states.
- Notice that the Additional Processing plug-ins create and complete a ToDo entry when a case enters and exits this state, respectively.

### Approved Formal Appeal

The following is an example of the configuration of the **Approved** status for formal appeal cases.

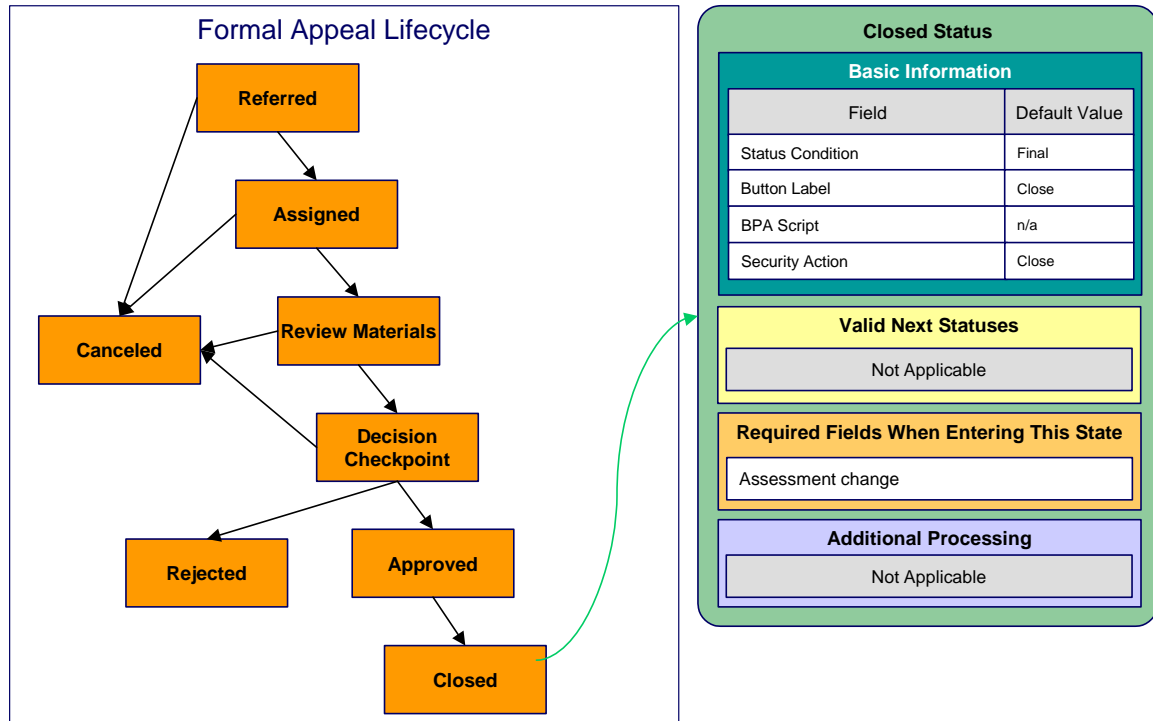


Note the following about this status:

- **Approved** is an interim state (meaning that it's not an initial or final state).
- It has a button label of **Approve Appeal**. This is the label on the button that a user presses to move a case into this state. This button will only appear on cases that are in the **Decision Checkpoint** state as this is the only state that can transition into the **Approved** state.
- We have referenced a BPA script that can assist a user in the steps required to adjust an assessment.
- We have associated the **Approve** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that Valid Next Statuses are what restricts a case in this state to be transitioned to the **Closed** and **Rejected** states.

### Closed Formal Appeal

The following is an example of the configuration of the **Closed** status for formal appeal cases.

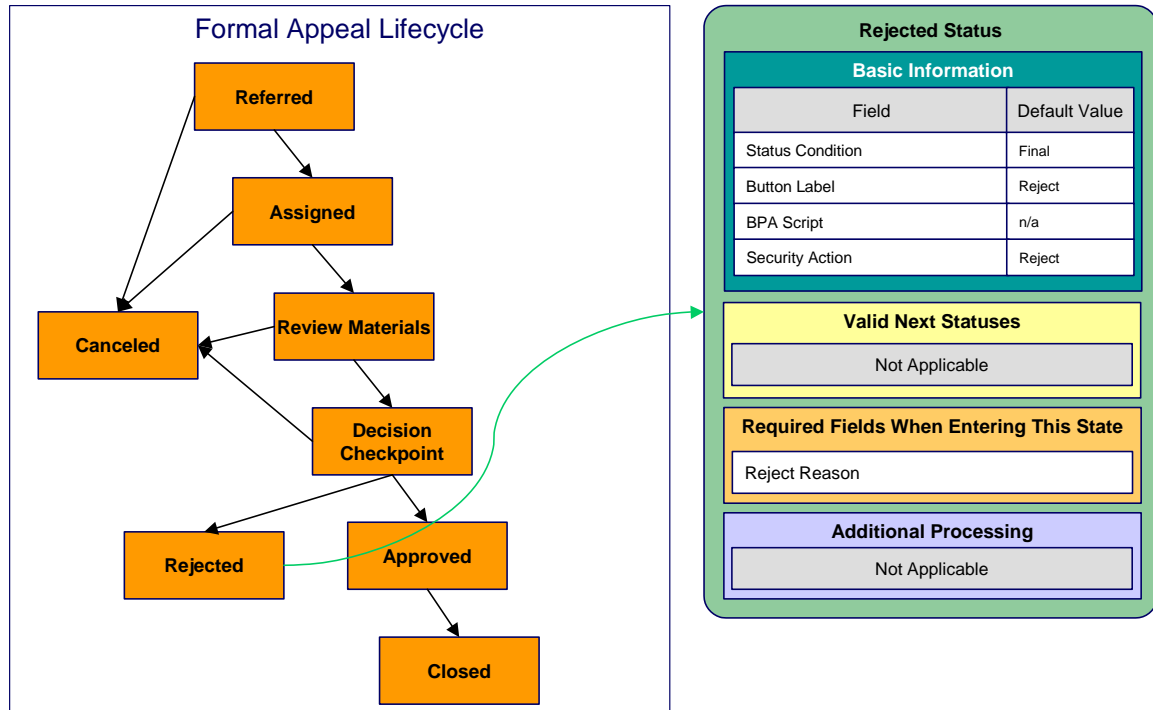


Note the following about this status:

- **Closed** is a final state (meaning that no further action is necessary).
- It has a button label of **Close**. This is the label on the button that a user presses to move a case into this state. This button will only appear on cases that are in the **Approved** state as this is the only state that can transition into the **Closed** state.
- We have not referenced a BPA script because this is a final state (and no additional user action is necessary).
- We have associated the **Close** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that there are no Valid Next Statuses (because this is a final state). If you wanted to allow a Closed case to be reopened, you would need to define the state into which a Closed case could be transitioned.
- Notice that the **Assessment Change** must be specified on the case before it can be moved into this state. This would be the difference to the taxpayer's balance after the assessment adjustment took place.

### Rejected Formal Appeal

The following is an example of the configuration of the **Rejected** status for formal appeal cases.

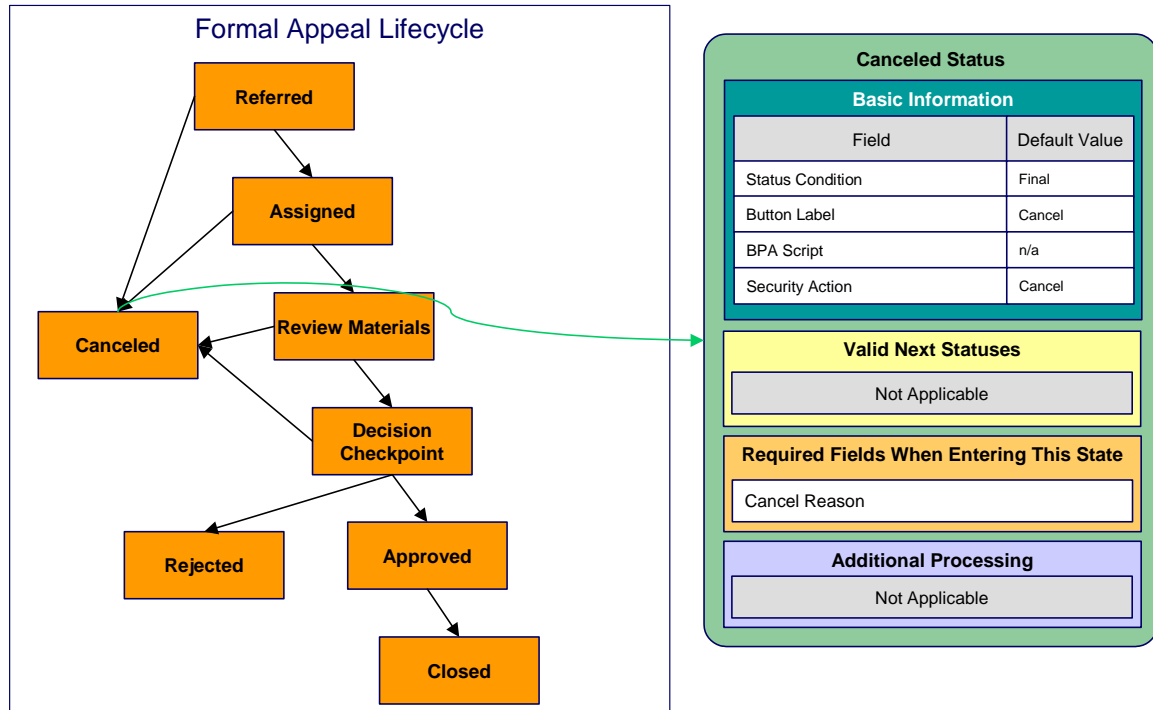


Note the following about this status:

- **Rejected** is a final state (meaning that no further action is necessary).
- It has a button label of **Reject**. This is the label on the button that a user presses to move a case into this state. This button will appear on cases that are in the **Decision Checkpoint** state.
- We have not referenced a BPA script because this is a final state (and no additional user action is necessary).
- We have associated the **Reject** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that there are no Valid Next Statuses (because this is a final state). If you wanted to allow a **Rejected** case to be reopened, you would need to define the state into which a **Rejected** case could be transitioned.
- Notice that a **Reject Reason** must be specified on the case before it can be moved into this state.

### Canceled Formal Appeal

The following is an example of the configuration of the **Canceled** status for formal appeal cases.



Note the following about this status:

- **Canceled** is a final state (meaning that no further action is necessary).
- It has a button label of **Cancel**. This is the label on the button that a user presses to move a case into this state. This button will appear on cases that are in the **Referred**, **Assigned**, **Review Materials** and **Decision Checkpoint** states.
- We have not referenced a BPA script because this is a final state (and no additional user action is necessary).
- We have associated the **Cancel** action with this status. This means that only users with rights to this action for the application service defined on the case type can move a case into this state.
- Notice that there are no Valid Next Statuses (because this is a final state). If you wanted to allow a **Canceled** case to be reopened, you would need to define the state into which a **Canceled** case could be transitioned.
- Notice that a **Cancel Reason** must be specified on the case before it can be moved into this state.

## Setting Up Case Management Options

The topics in this section describe how to set up the system to enable case management.

**Warning!** The following topics assume you thoroughly understand the concepts described under [The Big Picture Of Cases](#).

### Contents

[Installation Options](#)  
[Setting Up Application Services](#)  
[Setting Up Scripts](#)  
[Setting Up ToDo Types](#)  
[Setting Up Characteristic Types](#)  
[Setting Up Case Types](#)

## Installation Options

### Contents

[Case Info May Be Formatted By An Algorithm](#)  
[Alert Info Is Controlled By An Installation Algorithm](#)

### Case Info May Be Formatted By An Algorithm

An algorithm may be plugged in on the [installation record](#) to format the standard case information that appears throughout the system. Refer to [CSIN-DFLT](#) for an example of this algorithm.

This algorithm may be further overridden by a "Case information" plug-in on the [Case Type](#). Refer to [C1-CT-INFO](#) for an example of this algorithm.

### Alert Info Is Controlled By An Installation Algorithm

An algorithm that is plugged in on the [installation record](#) is responsible for formatting the alerts that highlight if the person / account / location in context has open cases. Refer to [CCAL-CASE](#) for an example of this algorithm.

## Setting Up Application Services

As described under [Access Rights](#), you can prevent unauthorized users from accessing cases. The following points describe how to implement this type of security:

- Create an [application service](#) for each case type that needs to be secured
- Create an action on the application service for each status you need to secure
- Link the valid [user groups](#) to the application service and define which actions they can perform
- Define the application service on the [case type](#)
- Define the related action for each status on the [case type / status](#)

## Setting Up Scripts

As described under [Scripts and Cases](#), BPA scripts can facilitate the creation and working of cases. Refer to the [Defining Script Options](#) for instructions describing how to set up scripts.

## Setting Up ToDo Types

As described under [ToDo's and Cases](#), ToDo entries can be used to highlight cases that require user attention.



The following points provide a high-level description of how to create (and complete) ToDo entries for a case type:

- Create a ToDo type for each different type of ToDo entry used during a case's lifecycle
  - On the ToDo type, think carefully about the roles whose users can work on the entries
  - Also consider if you would like a BPA script launched when a user selects the entry
- Specify the ToDo type on the appropriate [entry processing](#) or [exit processing](#) algorithm
- If you want the system to automatically complete ToDo entries, specify the ToDo type on the appropriate [entry processing](#) or [exit processing](#) algorithm

Please be aware that the case maintenance transaction highlights the number of open and being worked ToDo entries linked to the case being displayed on the page. However, the system can only do this if the ToDo entries reference a [foreign-key characteristic](#) whose foreign key references the case table. If you use the [CSEN-TD](#) algorithm to create ToDo entries when a case enters a given state, this algorithm will do this for you if:

- You have set up a [foreign-key characteristic type](#) whose [foreign key](#) references the case table
- In addition, the characteristic type must reference a characteristic entity of ***ToDo Entry***

## Setting Up Characteristic Types

As described under [Additional Information](#), some of your cases may require additional information (in the form of [characteristics](#)). If this is true, you must set up the characteristic types before setting up the case types.

Refer to [Setting Up ToDo Types](#) for instructions regarding a characteristic type that must be set up in order for the system to know the ToDo entries that are associated with a case.

If you use the [CSEN-CC](#) algorithm to create customer contacts when a case enters a given state, you should set up a [foreign-key characteristic type](#) as follows:

- Its [foreign key](#) must reference the case table
- In addition, the characteristic type must reference a characteristic entity of ***Customer Contact***

## Setting Up Case Types

The case type maintenance transaction is used to maintain your case types. The topics in this section describe how to use this transaction.

Refer to [The Big Picture Of Cases](#) for more information about how a case type encapsulates the business rules that govern a case.

### Contents

- [Case Type - Main](#)
- [Case Type - Case Characteristics](#)
- [Case Type - Lifecycle](#)

## Case Type - Main

Use this page to define basic information about a case type.

Open the case type page by selecting **Case Type** from the admin menu.

**Recommendation.** Before using this transaction, we strongly recommend that you review the [Examples of Case Types](#).

### Main Information

Enter a unique **Case Type** code and **Description** for the case type.

Use **Long Description** to provide a more detailed explanation of the purpose of the case type.

**Person Usage** controls the applicability of a person on cases of this type. Select **Required** if a person must be defined on this type of case. Select **Optional** if a person can optionally be defined on this type of case. Select **Not Allowed** if a person is not allowed on this type of case.

**Account Usage** controls the applicability of an account on cases of this type. Select **Required** if an account must be defined on this type of case. Select **Optional** if an account can optionally be defined on this type of case. Select **Not Allowed** if an account is not allowed on this type of case.

**Location Usage** controls the applicability of a location on cases of this type. Select **Required** if a location must be defined on this type of case. Select **Optional** if a location can optionally be defined on this type of case. Select **Not Allowed** if a location is not allowed on this type of case.

If you need to restrict access to cases of this type to specific user groups, reference the appropriate **Application Service**. Refer to [Setting Up Application Services](#) for the details of how to secure access to your cases.

If you are configuring a case type to handle the processing of data defined via a **Business Object**, associating the case type with a business object serves to link the properties of the business object (e.g. BO options) with cases of that type. Refer to [Business Objects](#) for further information. In addition, refer to [Automatic Transition Rules](#) for information on the role of BO options in case auto-transition errors.

**Responsible User Usage** controls the applicability of a responsible user on cases of this type. Select **Required** if a responsible user must be defined on this type of case. Select **Optional** if a responsible user can optionally be defined on this type of case. Select **Not Allowed** if a responsible user is not allowed on this type of case. Refer to [Responsible User Applicability](#) for more information.

### Contact Information Fields

There are three contact information fields: **Contact Person & Method Usage**, **Contact Instructions Usage**, and **Callback Phone Usage**. These fields are used to determine whether or not each type of contact information must be entered on case records with this case type. Select **Required** if the contact information must be entered, select **Optional** if the user can choose whether or not to include the contact information on this type of case, or select **Not Allowed** if the contact information cannot be entered on this type of case.

### Algorithms

The **Algorithms** grid contains algorithms that control functions for cases of this type. You must define the following for each algorithm:

- Specify the **System Event** with which the algorithm is associated (see the table that follows for a description of all possible events).
- Specify the **Sequence Number** and **Algorithm** for each system event. You can set the **Sequence Number** to 10 unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event**.

System Event	Optional / Required	Description
<i>Case Information</i>	Optional	<p>We use the term "Case information" to describe the basic information that appears throughout the system to describe a case. The data that appears in "case information" is constructed using this algorithm.</p> <p>Plug an algorithm into this spot to override the "Case information" algorithm on installation options or the system default "Case information" if no such algorithm is defined on installation options.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>

### Case Type Tree

The tree summarizes the case type's lifecycle. You can use the hyperlinks to transfer you to the **Lifecycle** tab with the corresponding status displayed.

### Case Type - Case Characteristics

To define characteristics that can be defined for cases of this type, open **Admin Menu, Case Type** and navigate to the **Case Characteristics** tab.

### Description of Page

Use **Sequence** to control the order in which characteristics are defaulted. Turn on the **Required** switch if the **Characteristic Type** must be defined on cases of this type. Turn on the **Default** switch to default the **Characteristic Type** when cases of this type are created. Enter a **Characteristic Value** to use as the default for a given **Characteristic Type** when the **Default** box is checked. Refer to [Required Fields Before A Case Enters A State](#) for a description of how you can make option characteristics required at later stages in a case's lifecycle.

### Case Type - Lifecycle

Case types that involve multiple users and multiple potential outcomes have complex lifecycles. Before you can design a case type's lifecycle, it's important that you thoroughly understand the concepts described under [Lifecycle](#) and [Status-Specific Business Rules](#). After thoroughly understanding these concepts, we recommend you perform the following design steps:

- Draw a "state transition diagram" as illustrated above under [Lifecycle](#). Keep in mind that if your state transition diagram is complex, your cases will be complex. While some cases warrant complexity, you should always ask yourself if there aren't better ways to achieve the desired results if your first effort results in complexity.
- Determine which characteristics (if any) are required during each stage of a case's lifecycle

- Determine when ToDo entries (if any) should be created (and completed) during a case's lifecycle
- Determine additional validation (if any) that should be executed before a case enters and exits each state
- Determine additional processing (if any) that should transpire when a case enters or exits each state
- Determine if scripts are warranted to help users work the cases and, if so, design the scripts for each applicable state

When the above tasks are complete, you will be ready to set up a case type's lifecycle.

Open the Lifecycle page by selecting **Case Type** from the admin menu then the **Lifecycle** tab.

**Note.** You can navigate to a status by clicking on the respective node in the tree on the Main tab. You can also use the hyperlinks in the Next Statuses grid to display a specific status in the accordion.

### Main Information

The **Status** accordion contains an entry for every status in the case type's [lifecycle](#).

Use **Status** to define the unique identifier of the status. This is NOT the status's description, it is simply the unique identifier used by the system.

Use **Description** to define the label that appears on the lifecycle accordion as well as the status displayed on the case.

Use **Script** to reference a BPA script that can assist a user work on a case while it's in this status. Refer to [A Script That Helps A User Work Through A Case](#) for the details.

Use **Access Mode** to define the action associated with this status. This field is disabled if an application service is not specified on the Main page. Refer to [Access Rights](#) for the details of how to use this field to restrict which users can transition a case into this state.

Use **Batch** to specify a batch control that will auto-transition the case. Any case in a status configured with a batch control will be transitioned when the batch job runs (rather than when [CASETRAN](#) is executed). For this purpose, batch process [C1-CSTRS \(Case Scheduled Transition\)](#) is supplied with base package, which will execute all Exit Status logic for the current status, and Enter Status logic for the destination status. You may choose to create a batch process with your own transition logic.

**Note.** If you wish to defer transitioning a case in a particular status until the batch process on your case type status is executed, you should not populate an Auto-Transition algorithm on that status. Otherwise, CASETRAN will transition the case according to your Auto-Transition logic.

Use **Comment** to describe the status. This is for your internal documentation requirements.

Use **Sequence** to define the relative order of this status in the tree on the Main page.

Use **Status Condition** to define if this status is an *Initial*, *Interim* or *Final* state. Refer to [One Initial State and Multiple Final States](#) for more information about how this field is used.

Use **Transitory State** to indicate whether a case should ever exist in this state. Only **Initial** or **Interim** states can have a transitory state value of **No**.

The **Alert Flag** is used to indicate whether or not an alert should be displayed for taxpayers with cases in the state. (The alert is shown via the base package Installation – Alert algorithm.)

### Algorithms

The **Algorithms** grid contains algorithms that control important functions for cases of this type. You must define the following for each algorithm:

- Specify the **System Event** with which the algorithm is associated (see the table that follows for a description of all possible events).
- Specify the **Sequence Number** and **Algorithm** for each system event. You can set the **Sequence Number** to 10 unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event** (note, all system event's are optional and you can define an unlimited number of algorithms for each event).

System Event	Description
<i>Auto Transition</i>	This algorithm is executed to determine if a case that's in this state should be transitioned into another state. Refer to <a href="#">Automatic Transition Rules</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Enter Processing</i>	This algorithm holds additional processing that is executed when a case is transitioned into this state. You can also specify state transition logic within Enter Processing routines. Refer to <a href="#">Additional Processing When Entering A State</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Enter Validation</i>	This algorithm holds validation logic that executes before a case can enter a given state. Refer to <a href="#">Validation Before A Case Enters A State</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Exit Processing</i>	This algorithm holds additional processing that is executed when a case is transitioned out of this state. Refer to <a href="#">Additional Processing When Exiting A State</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Exit Validation</i>	This algorithm holds validation logic that executes before a case can be transitioned out of a given state. Refer to <a href="#">Validation Before A Case Exits A State</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Script Launching</i>	This algorithm sets the script launching option for the script associated with a given state, if any. Refer to <a href="#">Script Launching Option</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.

### Next Statuses

Use the **Next Statuses** grid to define the statuses a user can transition a case into while it's in this state. Refer to [Valid States versus State Transition Rules](#) for more information. Please note the following about this grid:

- Use **Action Label** to indicate the verbiage to display on the action button used to transition to this status.

- **Sequence** controls the order of the buttons that appear on [Case - Main](#). Refer to [Buttons Are Used To Transition A Case Into A State](#) for more information.
- **Use as Default** controls which button (if any) is the default button.
- **Transition Condition** may be configured to identify a common transition path for cases of this type in the current state. This transition condition may then be referenced across multiple case types. You'll need to add values to Look Up table field **TR\_COND\_FLG** that fit the typical transitions for your case types (e.g. **Ok**, **Error**, etc.).

By assigning the transition condition value to a given "next status", you can design your Enter State transition or Auto-Transition logic to utilize those flag values *without specifying a status particular to a given case type*. Thus, similar logic may be used across a range of case types to transition a case into, for example, the next **Ok** state for the case's current status.

- **Transition Role** controls whether only the **System** or both **System and User** have the ability to transition a case into a given "next status".
- You can use the status description hyperlink to open the Status accordion to the respective status.
- When you initially set up a case type, none of the statuses will reside on the database and therefore you can't use the search to define a "next status". We recommend working as follows to facilitate the definition of this information:
  - Leave the Next Statuses grid blank when you initially define a case type's statuses
  - After all statuses have been saved on the database, update each status to define its Next Statuses (this way, you can use the search to select the status).

### Required Characteristics

Use the **Required Characteristics** grid to define characteristics that are required when a case enters this state. Only **Optional** characteristics defined on the main page appear in this grid. Refer to [Required Fields Before A Case Enters A State](#) for more information.

# Reports Addendum

This chapter is an addendum to the general [Defining and Designing Reports](#) chapter. This addendum describes the sample reports that are provided with Oracle Enterprise Taxation Management.

## Description of Sample Reports

This section provides an overview of each sample report supplied with Oracle Enterprise Taxation Management that may be found in the demonstration database. They may be used by your organization as they are or as a starting point for creating a [new report](#).

**Account Security.** Please note that the sample reports provided with the product do NOT incorporate account security. If a user has been given security to view the report, then all the data in the report is available for viewing.

### Contents

- [Billed Revenues by Rate - CI\\_BILREV](#)
- [Case Statistics By Case Type - CI\\_CSESTS](#)
- [Case Statistics for a Given Status - CI\\_CSESGS](#)
- [Customer Contacts by Type - CI\\_CUSTCN](#)
- [GL Accounting Summary - CI\\_GLACSM](#)
- [Open Cases By Type - CI\\_CSEOPN](#)
- [Payments Balance - CI\\_PMTBAL](#)
- [Receivables Aging - CI\\_RCVAGA](#)
- [Tax Payables Analysis - CI\\_TXPYBL](#)
- [To Do Entries - CI\\_TDENTR](#)

## Billed Revenues by Rate - CI\_BILREV

### Parameters

Parameter	Parameter Code	Description
Accounting Period	P_ACCT_PERIOD	Defines the accounting period used for the report. A valid fiscal year and accounting period for a valid accounting calendar must be provided.
Account type Characteristic	P_CHAR_TYPE	Defines a Characteristic Type for a characteristic linked to the Distribution Code to define an account type.
Account type	P_REV_ACCTY_CHAR	Account type Char Value for Revenue related GL Accounts. The char type defined for this parameter should match the Char Type code defined as parameter #2. The parameter value indicated for this parameter should be one that represents revenue accounts.

### Report Description

This is an analysis report for the billed revenues for an accounting period according to the various rates, which were in effect in the system. The information in this report helps to adjust rates in order to achieve better financial results and comply with regulations and market trends.

This report selects all records in the financial transaction GL collection that satisfy the following criteria:

- The financial transaction is frozen.
- The Accounting Date on the financial transaction within input accounting period (parameter 1)
- The distribution code associated with the GL entry has a characteristic type and value that matches the input account type Characteristic and account type (parameters 2 & 3)

**Performance Consideration.** If your implementation chooses to use this report, you may consider adding an index to the CI\_FT table on ACCOUNTING\_DT to aid in performance. When making this decision, carefully weigh the benefit of improving report performance against the possible degradation to the performance of day-to-day processing as a result of defining a new index. Note that many companies opt to create a reporting database that is a shadow of production to ensure that indexes defined to benefit reports may be created without any affect on the production environment.

## Case Statistics By Case Type - CI\_CSESTS

### Parameters

Parameter	Parameter Code	Description
Start Date	P_FROM_DT	See the report's description for how this field is used. If start date is not specified, it is defaulted to 7 days prior to the end date.
End Date	P_TO_DT	See the report's description for how this field is used. If end date is not specified, it is defaulted to the current processing date.
Case Condition (Open, Closed)	P_COND_FLG	If specified, only cases in this condition are included in the report. If left blank, the reports produces statistics for both open and closed cases.

### Report Description

This report provides two types of statistics:

1. Open cases whose creation date falls between the input Start Date and End Date (inclusive)
2. Closed cases whose closing date falls between the input Start Date and End Date (inclusive)

The third parameter is only used if you want to restrict the statistics to only open or closed cases. If you leave this parameter blank, both open and closed statistics will be produced.

The following information is provided in graphical format:

- Number of cases by case type
- Percentage of cases by case type



## Case Statistics for a Given Status - CI\_CSESGS

### Parameters

Parameter	Parameter Code	Description
Start Date	P_FROM_DT	See the report's description for how this field is used. If start date is not specified, it is defaulted to 7 days prior to the end date.
End Date	P_TO_DT	See the report's description for how this field is used. If end date is not specified, it is defaulted to the current processing date.
Case Type/Status	P_CASE_STATUS_CD	This is the desired Case Type and Status that will be reported on.
Responsible User	P_CASE_OWNER	If specified, only cases with this responsible user are included in the report.
First Bucket High Limit	P_B1_LIMIT	Cases that took <= this number of days to reach the given status will be grouped together for statistical reporting.
Second Bucket High Limit	P_B2_LIMIT	Cases that took <= this number of days but more than the first bucket high limit to reach the given status will be grouped together for statistics reporting.
Third Bucket High Limit	P_B3_LIMIT	Cases that took <= this number of days but more than the second bucket high limit to reach the given status will be grouped together for statistics reporting. Cases that took more than this number of days are included in another group.

### Report Description

This report shows cases of a given case type that transitioned to a given status during a given date range.

Graphs are printed to show the number and percentage of cases grouped by the time it took to reach the status. These statistics are grouped into age buckets whose size is controlled by the last 3 parameters.

Summary statistics are also printed showing the minimum, maximum, average and median times for these cases.

## Customer Contacts by Type - CI\_CUSTCN

### Parameters

Parameter	Parameter Code	Description
Start Date	P_FROM_DT	Start date to use for reporting customer contacts. If not defined, the start date is set to the current date minus 7 days.
End Date	P_TO_DT	End date to use for reporting customer contacts. If not defined, End Date is set to the current date.

Customer Contact Class / Type	P_CC_TYPE_CD	Specify a Customer Contact Class / Type to restrict the report output to a specific class / type.
-------------------------------	--------------	---

### Report Description

This report lists all customer contacts in the system created within the input date range. You may optionally restrict the report to customer contacts for a given Customer Contact Class / Type (parameter 3).

**Graphs.** The information on this report is shown in both textual and graphical formats.

**Performance Consideration.** If your implementation chooses to use this report, you may consider adding an index to the CI\_CC table on CC\_DTTM to aid in performance. When making this decision, carefully weigh the benefit of improving report performance against the possible degradation to the performance of day-to-day processing as a result of defining a new index. Note that many companies opt to create a reporting database that is a shadow of production to ensure that indexes defined to benefit reports may be created without any affect on the production environment.

## GL Accounting Summary - CI\_GLACSM

### Parameters

Parameter	Parameter Code	Description
Accounting Period	P_ACCT_PERIOD	Defines the accounting period used for the report. A valid fiscal year and accounting period for a valid accounting calendar must be provided.
Account Type Characteristic	P_CHAR_TYPE	Defines a Characteristic Type for a characteristic linked to the Distribution Code to define an account type. The account types for the GL accounts are used for grouping the output to the report.

### Report Description

This is a financial audit report used to check the financial details in Oracle Enterprise Taxation Management for an accounting period against the GL system. The report summarizes all financial transaction (FT) information for a given accounting period according to the different operating and GL divisions and according to various levels of the account GL information.

**Performance Consideration.** If your implementation chooses to use this report, you may consider adding an index to the CI\_FT table on ACCOUNTING\_DT to aid in performance. When making this decision, carefully weigh the benefit of improving report performance against the possible degradation to the performance of day-to-day processing as a result of defining a new index. Note that many companies opt to create a reporting database that is a shadow of production to ensure that indexes defined to benefit reports may be created without any affect on the production environment.

## Open Cases By Type - CI\_CSEOPN

### Parameters

Parameter	Parameter Code	Description
Start Date	P_FROM_DT	See the report's description for how this field is used. If start date is not specified, it is defaulted to 7 days prior to the end date.
End Date	P_TO_DT	See the report's description for how this field is used. If end date is not specified, it is defaulted to the current processing date.
Case Type	P_CASE_TYPE_CD	If specified, only cases of this type are included in the report.
Responsible User	P_CASE_OWNER	If specified, only cases with this responsible user are included in the report.
First Bucket High Limit	P_B1_LIMIT	Cases that are open for less than or equal to this number of days will be grouped together for statistical reporting.
Second Bucket High Limit	P_B2_LIMIT	Cases that are open less than or equal to this number of days but more than the first bucket high limit will be grouped together for statistics reporting.
Third Bucket High Limit	P_B3_LIMIT	Cases that are open less than or equal to this number of days but more than the second bucket high limit will be grouped together for statistics reporting. Cases that took more than this number of days are included in another group.

### Report Description

This is a report on open cases that were created between a given date range.

The report can be limited to a specific type and/or responsible user.

For each case type, the report shows the following:

- Number of open cases by age bucket (the last 3 parameters control the size (in days) of each bucket)
- Percentage of open cases by age bucket
- Details of the open cases

## Payments Balance - CI\_PMTBAL

### Parameters

Parameter	Parameter Code	Description
Start Date	P_FROM_DT	Report gets all the payments that have been received during a given date range (from and to date parameters). If start date is not defined by the user, it is set to 7 days prior to the current date.

End Date	P_TO_DT	End date of the date range. If not defined by user it is set to the current date
----------	---------	--

### Report Description

This report provides an overall view of all payments created within the input date range. It is typically used for financial control and audit purposes. The report provides summary information about valid payments received and about canceled payment. Data is summarized by the tender source and the type of payment.

## Receivables Aging - CI\_RCVAGA

### Parameters

Parameter	Parameter Code	Description
Cutoff Date	P_CUTOFF_DATE	The date from which the arrears buckets are calculated. If no value is entered, the default is the current date minus 7 days.
1 <sup>st</sup> Bucket High Limit	P_B1_LIMIT	High limit of 1st bucket.
2 <sup>nd</sup> Bucket High Limit	P_B2_LIMIT	High limit of 2 <sup>nd</sup> bucket.
3 <sup>rd</sup> Bucket High Limit	P_B3_LIMIT	High limit of 3 <sup>rd</sup> bucket.

### Report Description

This report lists all accounts and their arrears information as of the input cutoff date using a balance forward accounting method.

Outstanding debt is placed into the buckets provided as input using the age of the debt as of the cutoff date. Credits are applied to the oldest debt first. For each account a separate bucket is used to display new charges. In addition, the total accounts receivable balance is displayed for each account.

**Performance Consideration.** If your implementation chooses to use this report, you may consider adding an index to the CI\_FT table on ARS\_DT to aid in performance. When making this decision, carefully weigh the benefit of improving report performance against the possible degradation to the performance of day-to-day processing as a result of defining a new index. Note that many companies opt to create a reporting database that is a shadow of production to ensure that indexes defined to benefit reports may be created without any affect on the production environment.

## Tax Payables Analysis - CI\_TXPYBL

### Parameters

Parameter	Parameter Code	Description
Start Date	P_FROM_DT	Show summary of the tax amounts starting from this date. If not specified, the system will default this value to the current date minus 7 days.
End Date	P_TO_DT	Show summary of the tax amounts up to this date.

		If not specified, the system will default this value to the current date.
Account type Characteristic	P_CHAR_TYPE	Defines a Characteristic Type for a characteristic linked to the Distribution Code to define an account type.
Account type	P_TAX_ACCTY_CHAR	Account type Char Value for tax related GL Accounts. The char type defined for this parameter should match the Char Type code defined as parameter #3. The parameter value indicated for this parameter should be one that represents tax liability accounts.

### Report Description

This report displays a summary of the tax amounts that were levied by the company to taxpayers within the input date range. It also includes the tax exemption information for that period.

This report select all records in the financial transaction GL collection that satisfy the following criteria:

- The financial transaction is frozen.
- The Accounting Date on the financial transaction within the input date range
- The distribution code associated with the GL entry has a characteristic type and value that matches the input account type Characteristic and account type (parameters 3 & 4)

The report also provides tax exemption information for bill segments whose financial transactions satisfy the above criteria. The tax exemption information is retrieved by looking at the bill calculation lines associated with the FT's bill segment.

**Performance Consideration.** If your implementation chooses to use this report, you may consider adding an index to the CI\_FT table on ACCOUNTING\_DT to aid in performance. When making this decision, carefully weigh the benefit of improving report performance against the possible degradation to the performance of day-to-day processing as a result of defining a new index. Note that many companies opt to create a reporting database that is a shadow of production to ensure that indexes defined to benefit reports may be created without any affect on the production environment.

## To Do Entries - CI\_TDENTR

### Parameters

Parameter	Parameter Code	Description
ToDo Entry Status	P_ENTRY_STATUS_FLG	Defines if the ToDo entries on the report should be limited to those with a given status value. If this parameter is left blank, the report will show all <i>open</i> and <i>being worked</i> ToDo entries.
ToDo Type	P_TD_TYPE_CD	Defines if the ToDo entries on the report should be limited to those of a given ToDo type. If this parameter is left blank, the report will show all ToDo type that have at least one <i>open</i> or <i>being worked</i> entry.

**Report Description**

The report shows open and being worked To Do entries.

You can limit the report to entries in a given status by specifying the desired **To Do Status** (open or being worked). If you don't specify a status, all *open* and *being worked* To Do entries will appear on this report.

You can also limit the report to entries of a given To Do Type by specifying the desired **To Do Type**. If you don't specify a To Do Type, all To Do Types with at least one entry in the *open* / *being worked* state will appear on this report.

**Graphs.** The information on this report is shown in both textual and graphical formats.

# Security Addendum

This chapter is an addendum to the general [Defining Security and User Options](#) chapter. This addendum describes security functionality that is specific to Oracle Enterprise Taxation Management.

## Contents

[Implementing Account Security](#)  
[Masking Sensitive Data](#)

## Implementing Account Security

**Important!** This section assumes you understand [The Big Picture of Row Security](#).

When you create an account, you must define which users can access the account's information. For example,

- If you have taxpayers in two geographic territories, you may need to restrict access to accounts based on the office that manages each territory. For example, only users in the northern office may manage accounts in the northern territory.
- If you have businesses and individual taxpayers, you may need to restrict access to these different taxpayer segments based on the skill set of the users. For example, some users are skilled in dealing with business taxpayers, while others are skilled in dealing with individual taxpayers.

By granting a user access rights to an account, you are actually granting the user access rights to the account's bills, payment, adjustments, etc.

Refer to [If You Do Not Practice Account Security](#) for setup instructions if your organization doesn't practice account security.

**Account security may also affect persons and locations.** Refer to [Persons Can Also Be Secured](#) for how access to person information is also restricted by account security. Refer to [Locations Can Also Be Secured](#) for how access to location information is also restricted by account security.

The topics in this section describe how to implement account security.

## Contents

[Persons Can Also Be Secured](#)  
[Locations Can Also Be Secured](#)  
[Data Becomes Invisible When Access Is Restricted](#)  
[Restricted Transactions](#)  
[Account Security Case Studies](#)  
[The Default Access Group](#)  
[If You Do Not Practice Account Security](#)

## Persons Can Also Be Secured

It's important to be aware that persons can also be secured as a result of "account security". It works like this:

- If a person is linked to at least one account, users will not be allowed access to the person (or the person's related information) unless they have access to at least one of the person's accounts.
- If a person is not linked to any accounts (a rare situation), any user may access the person.

**How are persons linked to accounts?** A person is linked to an account when an account is created using the method described under [How To Add A New Taxpayer From Control Central](#). In addition, you may manually link and unlink persons from account using the [Account – Person](#) page.

## Locations Can Also Be Secured

It's important to be aware that locations can also be secured as a result of "account security". It works like this:

- If a location is linked to at least one account, users will not be allowed access to the location (or the location's related information) unless they have access to at least one of the location's accounts.
- If a location is not linked to an account (a rare situation), then all users may access the location.

## Data Becomes Invisible When Access Is Restricted

The following points summarize the impact of a user not having access to an account.

### Contents

[Account Security and Control Central](#)  
[Account Security and Searches \(and Maintenance Pages\)](#)  
[Account Security and ToDo Lists](#)

### Account Security and Control Central

This section summarizes the impact of account security on [Control Central](#):

- Searches are affected as follows:
  - An account will only be visible if a user has access to the account's access group.
  - Persons that are not linked to accounts will be visible to all users.
  - If a person is linked to an account, the person will only be visible if the user has access to at least one of the person's accounts.
  - Locations that are not linked to accounts will be visible to all users.
  - If a location is linked to an account, the location will only be visible if the user has access to at least one of the location's accounts.



- The alerts that highlight the existence of “multiple relationships” are not impacted by account security. Specifically:
  - The alert ***Person has multiple accounts*** will appear if the selected person is linked to multiple accounts, even if the user doesn’t have access to every account. Note well, the person couldn’t have been selected if the user didn’t have access rights to at least one account.
  - The alert ***Location has multiple accounts*** will appear if the selected location is linked to multiple account, even if the user doesn’t have access to every account. Note well, the location couldn’t have been selected if the user didn’t have access rights to at least one account.
- Only accounts to which the user has access will be displayed in the person tree.
- Only accounts to which the user has access will be displayed in the account tree.
- All other pages contain information related to Control Central’s current account context. The current account context can never reference an inaccessible account and therefore these pages are not impacted by account security.

### Account Security and Searches (and Maintenance Pages)

Searches are the gateway to the information that appears on maintenance pages. In general, account-related information is suppressed when a user doesn’t have access rights to the account. This suppression is true for rows that directly reference an account AND for rows that indirectly reference an account. For example:

- A user can only see bills associated with accounts to which they have access rights.
- A user can only see financial transactions associated with obligations that are, in turn, associated with accounts to which they have access rights.

**Person and location searches are also impacted.** Keep in mind that information will be suppress from both person and location-oriented searches if the person / location is related to accounts. Refer to [Persons Can Also Be Secured](#) for how access to person information is also restricted by account security. Refer to [Locations Can Also Be Secured](#) for how access to location information is also restricted by account security.

### Account Security and ToDo Lists

Account security does NOT impact the information that appears in a user’s ToDo list. Rather, we have assumed that your ToDo roles (and the users assigned to these roles) are consistent with your account security requirements. This can result in anomalies. For example, it’s possible for a supervisor to assign a bill segment error to a user who doesn’t have access to the bill segment’s account. This user will then see the related ToDo entry in their Bill Segments In Error ToDo list. However, when they drill down on the entry, account security will manifest itself (i.e., the user won’t be able to display the bill segment that’s in error). This happens because the drill down causes the bill segment search logic to execute. This logic inhibits the selection of bill segments if the user can’t access the related account.

To minimize these anomalies, we recommend the following:

- Setup [To Do Roles](#) consistent with your Data Access Roles.
- Setup [Account Management Groups](#) that are consistent with your Access Groups.

- Setup default To Do Roles on your Account Management Groups for each [ToDo type](#).

## Restricted Transactions

The following table lists all transactions that have some type of account security. The following notation is used to describe the type of account security:

- **Account-oriented.** This notation is used if the respective transaction uses basic account security (i.e., the user must belong to at least one data access role that has access to the account's access group in order to see the information).
- **Person-oriented.** This notation is used if the respective transaction uses person-oriented account security. Refer to [Persons Can Also Be Secured](#) for more information.
- **Location-oriented.** This notation is used if the respective transaction uses location-oriented account security. Refer to [Locations Can Also Be Secured](#) for more information.
- None of the above. Some unusual transactions have unusual implementations of account security. These are described below.

Transaction	Type of Account Security
Account	Account-oriented
Account Bill / Payment History	Account-oriented
Account Financial History	Account-oriented
Account Payment History	Account-oriented
Account Person Replicator	Account-oriented
Adjustment	Account-oriented
Adjustment Calculation Line Characteristics	Account-oriented
Bill	Account-oriented
Bill Print Group	Person-oriented
Bill Segment	Account-oriented
Billable Charge	Account-oriented
Case	Account-oriented, Person-oriented and Location-oriented
Collection Agency Referral	Account-oriented
Control Central	Account-oriented, Person-oriented and Location -oriented
Customer Contact	Person-oriented
Financial Transaction	Account-oriented
Financial Transaction on a Bill	Account-oriented
Financial Transaction on a Payment	Account-oriented
Match Event	Account-oriented
Multi-Cancel/Rebill	Account-oriented
Overdue Process	Account-oriented
Pay Plan	Account-oriented
Payment	Account-oriented

Transaction	Type of Account Security
Payment Event	The user must have access to ALL accounts linked to the payment event.
Payment Event QuickAdd	The user must have access to ALL accounts linked to the payment event(s).
Payment QuickAdd	Account-oriented
Payment / Tender Search	Account-oriented
Person	Person-oriented
Location	Location -oriented
Obligation Billing History	Account-oriented
Obligation Cash Accounting Balance	Account-oriented
Obligation Financial History	Account-oriented
Obligation	Account-oriented

## Account Security Case Studies

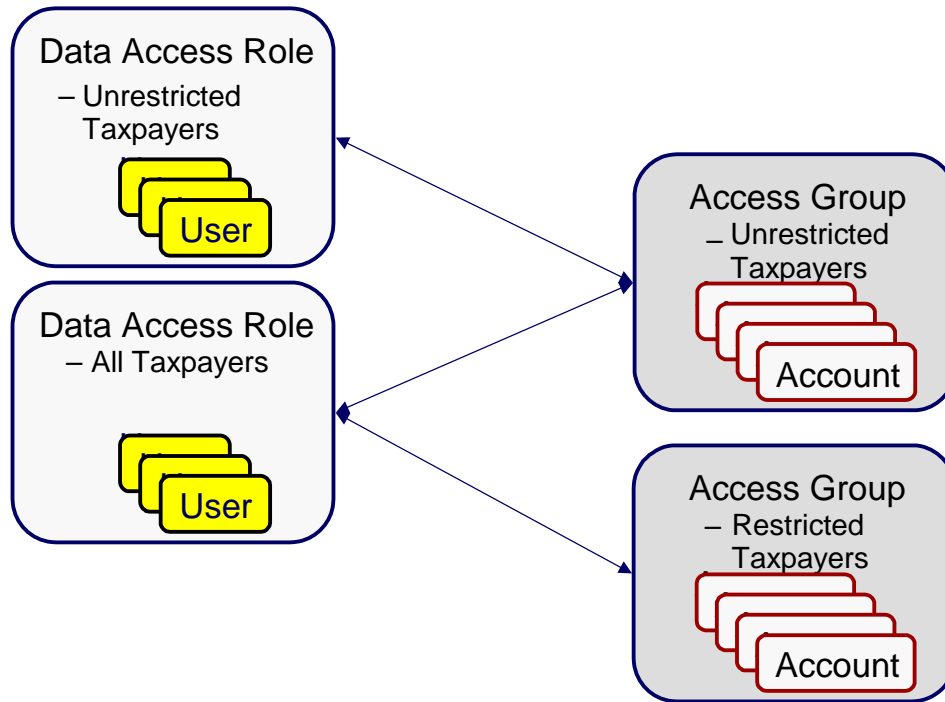
This section contains an example of how to implement account security. Use this example to form an intuitive understanding of these objects. Once this intuition is obtained, you'll be ready to design the account security objects for your own company.

### Securing Accounts Based On Account Type

Assume the following security requirement exists:

- You have two broad groups of accounts:
  - Unrestricted Taxpayer accounts for the general public.
  - Restricted Taxpayer accounts for individuals whose tax information is highly sensitive (politicians, celebrities, employees of the tax authority, etc.).
- Users can be classified as have one of the following access rights:
  - May access all accounts.
  - May only access the Unrestricted Taxpayer accounts.

The following diagram illustrates the access groups and data access roles required to implement these requirements:



Notice the following about the above:

- There are two access groups because access to accounts is based on whether the taxpayer's account is unrestricted or restricted.
- The Unrestricted Taxpayers data access role is only linked to the Unrestricted Taxpayers access group.
- The All Taxpayers data access role is linked to both the Unrestricted Taxpayers and Restricted Taxpayers access groups. Users with this role can therefore access all accounts.

## The Default Access Group

There are two ways an access group can be assigned to an account:

- The base package will default an account's access group based on the user who adds the account. It uses the user's [default access group](#) to do this.
- If you can conceive of a rule to assign an appropriate access group to a newly added account, you can have your programming staff introduce a user exit to the account row program to implement this rule. For example, a user exit could be developed that assigns an access group to an account based on its account type. The name of the user-exit is AFTER-MOVE-CORR-ADD and the name of the account row program is CIPCACCR.

**Warning!** Regardless of the method used to assign an account's access group, please be aware that the user who adds an account must have access to this access group.

**Subsequent changes to an account's access group.** A user may change an account's access group to any access group to which they have access.

## If You Do Not Practice Account Security

If you do not restrict access to accounts (i.e., all users can access all accounts), you must set up one access group and one data access role and then indicate all users are part of this role. You should also define the access group as the default access group on all of your users (so that new accounts are all labeled with this access group).

## Masking Sensitive Data

---

Refer to [Masking Data](#) for instructions describing how to configure the system to mask sensitive data like a taxpayer's social security number or bank account number. If your implementation intends to mask any of the information that appears in the [Taxpayer Information Zone](#), please navigate to this zone's documentation for special instructions.

# Defining Overdue Processing and Collection Options

The system periodically monitors how much your taxpayers owe to ensure they haven't violated your collection rules. When a violation is detected, the system initiates the appropriate activities (e.g., letters, collection agency referrals, and eventually write off). The topics in this section describe how to configure the system to manage your overdue processing and collection requirements.

**Collecting on unpaid debt.** The overdue processing module has been designed to collect on virtually anything from an unpaid bill to an unmatched financial transaction. You tell the system what you collect on by configuring the various overdue processing control tables.

**Straightforward rules = straightforward set up.** Setting up this module is as challenging as your organization's collection rules. If you have simple rules, the set up process will be straightforward. If your rules are complicated (e.g., they differ based on the type of taxpayer, the age of debt, the type of tax, etc.), your setup process will be more challenging.

Note: The terms "customer" and "taxpayer" may be used interchangeably in this chapter.

## Contents

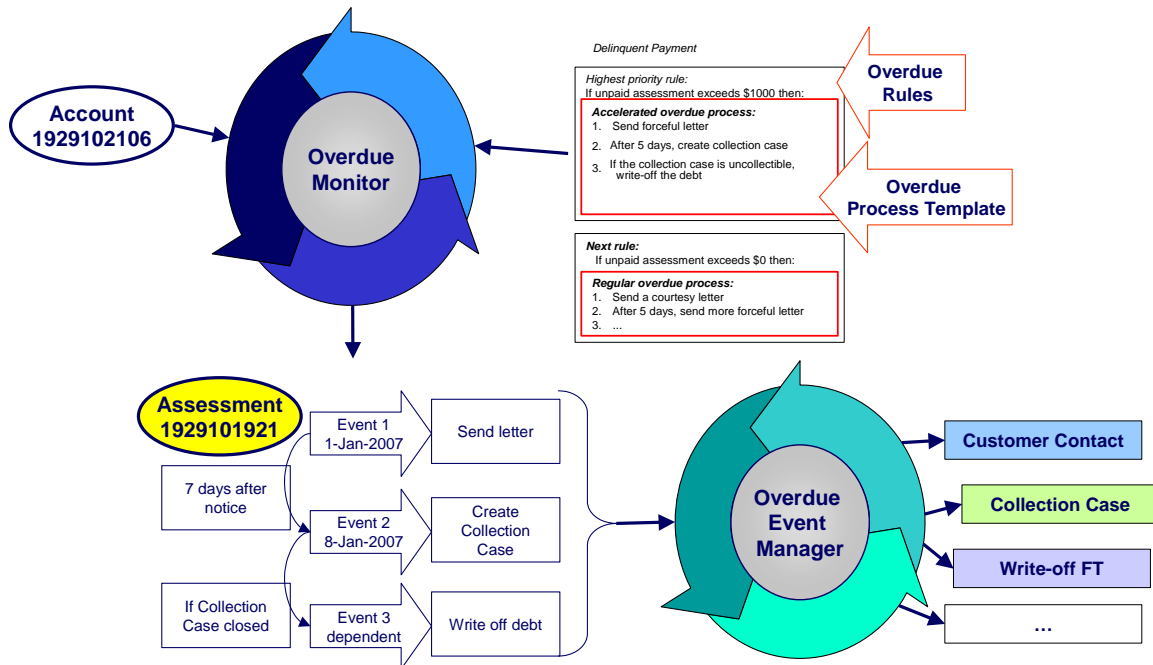
- [Case Study - Collecting On Overdue Debt](#)
- [How Does The Overdue Monitor Work?](#)
- [The Big Picture Of Overdue Processes](#)
- [Creating Overdue Procedures](#)

## Case Study - Collecting On Overdue Debt

The following topics introduce a case study that describes how overdue processing can be used to collect on overdue debt. This is just an example as virtually every aspect of overdue processing is configurable. Use this case study to familiarize yourself with the overdue processing core concepts.

## Monitoring Overdue Debt

The following diagram illustrates the objects and processes involved with collecting overdue debt.



There are many important concepts illustrated above:

**The Overdue Monitor checks if your accounts have debt that violates your overdue rules**

The [Overdue Monitor](#) is a background process that periodically reviews your account's financial obligations.

Note well: every account belongs to a collection class. Collection classes are monitored by the Overdue Monitor. This chapter describes the Overdue Monitor.

**Overdue rules define when and how unpaid debt is collected**

An account's [collection class overdue rules](#) have algorithms that monitor an account's financial obligations. These algorithms are invoked by the Overdue Monitor when it's [time to review](#) an account's obligations.

These algorithms can contain any type of criteria.

When you set up a monitoring algorithm, you define the type of overdue process that should be created when overdue debt is detected. You do this by defining the appropriate "overdue process template".

**An overdue process template defines how to handle overdue debt**

An [overdue process template](#) contains one or more [overdue event types](#). These define the number and type of events that are created to prod the taxpayer to pay. For example, you might set up an overdue process template with event types to send a series of letters followed up by a call.

The overdue process template has contains

the rules defining [when events are activated](#).

The specific action that's performed by an overdue event is controlled by the **Activation** algorithm defined on its event type. Refer to [Overdue Event Type - Main](#) for a list of the various **Activation** algorithms delivered with the base product.

**Multiple objects can be associated with a single process**

The above diagram shows a single assessment linked to an overdue process. It should be noted that an overdue process is capable of referencing multiple assessments (or other objects).

Note well: while a single overdue process can reference many overdue objects, all such objects must be of the same type. For example, you cannot commingle bills and obligations under a single overdue process. The type of object managed by an overdue process is defined on its [overdue process template](#).

**If a taxpayer pays the debt, the overdue process is cancelled**

If the overdue debt is paid, the [overdue process is canceled](#) real-time. You control if and how an overdue process is cancelled by setting up the appropriate rules on the [overdue process template](#).

**The Overdue Event Manager activates and triggers overdue events**

The [Overdue Event Manager](#) is a background process that activates overdue events on the appropriate date. On this date, the event's **Activation** algorithm(s) are called.

This Overdue Event Manager also has the responsibility of recursively activating later events that are dependent on the completion of earlier events.

**Events can be activated real-time**

[Overdue Process - Main](#) has a button that allows users to activate (and recursively trigger) overdue events online / real-time. This means you don't have to wait for a batch job to activate events.

**Overdue events can wait for related activities to complete**

As described above, an overdue event's **Activation** algorithm can create virtually any object. What wasn't explained is that the event can be set up to [wait](#) for the ancillary object to finish before it completes. For example, an event can create a To Do entry and wait for it to complete before the next event is triggered. You can introduce plug-ins to create and wait on virtually any object.

While an overdue event is in the **Wait** state, the Overdue Event Manager monitors the



state of the related object(s). When the related object completes, the event is transitioned to the Complete state (thus triggering dependent overdue events). Please see [Some Events Wait For Something Before Completing](#) for more information.

## How Does The Overdue Monitor Work?

This section describes how the Overdue Monitor background process (batch control: **C1-ADMOV**) uses your overdue rules to collect overdue debt.

**Recommendation.** We recommend that you familiarize yourself with the concepts described in the [case studies](#) before reading this section.

### Contents

- [Different Overdue Rules For Different Taxpayers](#)
- [Overdue Rules Are Embodied In Algorithms](#)
- [When Is An Account Monitored?](#)
- [Collection Class Defines If And How Accounts Are Monitored](#)

## Different Overdue Rules For Different Taxpayers

The Overdue Monitor uses rules to control how it monitors an account's debt. The system allows you to define different rules for different combinations of collection class, division and currency code. For example,

- You may have different collection rules for different jurisdictions (i.e., divisions).
- You may have different collection rules for different taxpayer segments. You differentiate your taxpayers in respect of the overdue via the **collection class on the taxpayers' accounts**. An account's initial collection class is defaulted from its account type. You may override an account's collection class at will.
- You may have different criteria for every currency in which you work because the monitoring process always compares a taxpayer's debt against some value and this value must be denominated in the taxpayer's currency. A taxpayer's currency is defined using a **currency code on the account**.

## Overdue Rules Are Embodied In Algorithms

Your organization's overdue rules are defined in algorithms plugged in on [Collection Class Overdue Rules](#) (in the **Overdue Monitor Rule** system event). When the Overdue Monitor analyzes an account, it uses the rules associated with the account's collection class, division and currency code. To analyze an account, it simply invokes these algorithms in sequence order, i.e., the lower the sequence, the higher its priority.

An **Overdue Monitor Rule** algorithm has two responsibilities:

- it determines if an account violates its overdue rules,
- if so, it creates one or more overdue processes using an [overdue process template](#)

**Additional rules.** Your implementation can have an **Overdue Monitor Rule** that caters for credit balances on obligations. For an example of an algorithm that creates an overpayment process for an obligation in credit, please refer to the base algorithm type [C1-CC-INITOP](#).

## When Is An Account Monitored?

The Overdue Monitor determines if an account violates your overdue rules at least every X days, where X is defined on the [Account Type - Controls](#) associated with the account's account type and division (in the field Min Compliance Review Freq (Days)).

In addition, the Overdue Monitor examines an account's debt as follows:

- X days after an account's bill due date (X is defined in the field Compliance Review Grace Days on [account type control](#)).
- If a payment is canceled with a cancellation reason that indicates non-sufficient funds.
- If a [match event](#) is added, changed or deleted.

**Additional events.** Your implementation can have other events trigger the analysis of an account by the Overdue Monitor such as the freezing of an FT that affects the account balance. To do this, add logic to insert a row on the CI\_ADM\_RVW\_SCH table when the event occurs. This row simply references the account ID to be reviewed and the desired review date. For an example of triggering review at FT Freeze time, please refer to the base algorithm type [C1-CFTZ-CMRV](#).

## Collection Class Defines If And How Accounts Are Monitored

As described above, every account references a collection class. The collection class defines if and how its accounts are monitored. There are the following options:

- The accounts are monitored by the Overdue Monitor (this is described in this chapter).
- The accounts are not monitored for overdue debt.

## The Big Picture Of Overdue Processes

As described above, the Overdue Monitor subjects your accounts to overdue rules. If a rule is violated, an overdue process is created. The topics in this section provide background information about overdue processes.

### Contents

- [How Are Overdue Processes Created?](#)
- [The Components Of An Overdue Process](#)
- [Experimenting With Alternative Overdue Process Templates](#)
- [Overdue Process Information Is Overridable](#)
- [Original and Unpaid Amounts](#)
- [Overdue Processes Are Highlighted Elsewhere](#)

[How Are Overdue Processes Cancelled?](#)  
[Overdue Processes Are Created From Templates](#)  
[The Big Picture Of Overdue Events](#)  
[The Big Picture of Collection Cases](#)  
[The Big Picture Of Collection Agency Referrals](#)  
[Write Offs Are Implemented Using Overdue Events](#)  
[Calendar vs. Work Days](#)

## How Are Overdue Processes Created?

As described [above](#), the system creates an overdue process when an account violates your overdue rules. In addition, a user can manually create an overdue process at their discretion.

## The Components Of An Overdue Process

The following topics describe the major components of an overdue process.

### Contents

[Overdue Objects](#)  
[Overdue Events](#)  
[Overdue Log](#)

### Overdue Objects

When an overdue process is created, the system links the overdue object(s) to the process. For example, if an overdue bill is detected, the bill is linked to the overdue process.

When you set up an [overdue process template](#), you define the type of object it collects on by defining the [foreign key characteristic type](#) used to reference the object. For example, when you set up an overdue process template to collect on bills, you define a foreign key characteristic type that references the bill object.

### Overdue Events

An overdue process has one or more overdue events. These events are the actions designed to encourage the taxpayer to pay. For example, you might set up overdue events that:

- Send letters (via the creation of a customer contact)
- Create To Do entries
- Impact the account's compliance rating
- Create a collection case to manage actions such as creating pay plans and referring debt to collection agencies
- ... (the list is only limited by your imagination as algorithms are used to perform the event's actions)

You define the number and type of events by configuring [overdue process templates](#). When the system creates an overdue process, it copies the events defined on the specified template.

It's important to note that all overdue events are created when the overdue process is created. A separate background process, the [Overdue Event Manager](#), is responsible for activating, monitoring, and triggering overdue events. Activation of an event causes the system to do whatever the event indicates; for instance, send a letter, send a To Do entry to a user or write-off debt.

## Overdue Log

Every overdue process has a log holding its history. Entries are added to the log when meaningful events occur, for example:

- When the process is created, a log entry is created to describe why the process was started.
- When an overdue event is activated, a log entry is created. These entries frequently contain a foreign key to the object that the event created so that users can easily drill down to the object from the log. For example, if an event creates a To Do entry, the To Do entry's foreign key is placed on the log and this allows a user to drill down on the log entry to see the To Do entry.
- When a process is canceled, a log entry is created to describe the circumstances of the cancellation (e.g., manual versus automated).
- Users can manually add log entries (you might want to think of these as "diary" entries) as desired.

Many of the base-product algorithms involved in overdue processing insert log entries so that a thorough audit trail is maintained. These algorithms have been designed to allow you to control the verbiage in each log entry by defining the desired message number using an algorithm parameter.

The log is viewable on the [Overdue Process - Log](#) page.

**More than just an audit trail.** Please note that the log entries are more than just an audit trail. The system makes use of the log entries to know what it did. For example, when an overdue event needs to monitor the state of the To Do entries that it created, it uses the log to determine the identity of these To Do entries.

## Experimenting With Alternative Overdue Process Templates

The system allows you to determine the efficacy of proposed overdue process templates using a small subset of taxpayers before implementing the templates on the entire taxpayer base. We use the term "champion / challenger" to reference this functionality.

We'll use an example to explain. Let's assume your prevailing overdue process template for individual taxpayers starts with a "gentle reminder" letter followed 10 days later by a letter threatening to place a lien to secure the debt if payment is not received. You may want to experiment with the impact of a change to this template. For example, you may want to change the "gentle reminder" to something more assertive and follow this up 5 days later with an even sterner warning. You can use the "champion / challenger" functionality to perform this experiment.

The following points describe how to implement "champion / challenger" functionality:

- Set up a "challenger" overdue process template for each template that you want to experiment with.
- Insert a new **Champion/Challenger** option on the [Overdue Processing Feature Configuration](#) for every champion template. Each option's value defines:
  - the "champion" overdue process template code

- the "challenger" overdue process template code
- the percentage of the time the system should use the "challenger" template

Keep in mind that you can only experiment with one challenger template per champion template.

After setting up the above, the [Overdue Rule Plug-In](#) will use the challenger template X% of the time rather than the champion template.

If you are using the Oracle Utilities Business Intelligence product, you can configure analytic zones in innumerable ways to compare the efficacy of the champion versus the challenger. For example,

- You can set up a graph to show the average duration of each type of process.
- You can set up a graph to show the average dollars that were successfully collected.
- You can set up a dimensional scorecard to show how each template performed in different regions (or account types or ...).
- Etc (the list is limited by your imagination)

## Overdue Process Information Is Overridable

"Overdue process info" is the concatenated string of information that summarizes an overdue process throughout the user interface. The base-product logic constructs this string by concatenating the following information:

- The description of its overdue process template
- Its status
- For **active** processes, the number of days since it was created. For **inactive** processes, the number of days since it was inactivated.
- For **active** processes, the unpaid amount of the objects being collected

If you'd prefer a different info string, you can develop a new algorithm and plug-it in on your [overdue process templates](#). This design allows some / all overdue process templates to have an override info string.

## Original and Unpaid Amounts

There are two amounts associated with each overdue object linked to an overdue process: its Original Amount and its Unpaid Amount. These amounts are used throughout overdue processing.

You control how these amounts are calculated by defining the appropriate algorithm on your [overdue process templates](#). For example, you can plug in a base-product algorithm ([C1-OBASM-AMT](#)) if you collect on overdue assessments or obligations. The base algorithm uses the **Obligation Type - Determine Detailed Balance** algorithm to calculate the amounts.

## Overdue Processes Are Highlighted Elsewhere

The topics in this section describe how other parts of the system highlight the existence of overdue processes.

## Contents

[Alert Zone](#)

[Compliance Zone](#)

[Account Activity History Zone](#)

## Alert Zone

If you plug-in the appropriate alert algorithm ([C1-OD-PROC](#)) on the Installation record, alert(s) will be shown for active overdue processes in the Alert Zone that appears in the Dashboard and on Control Central - Account Information.

## Compliance Zone

The Compliance zone on [Control Central - Account Information](#) shows **active** overdue processes.

## Account Activity History Zone

The Account Activity History Zone on [Control Central - Account Information](#) shows pending and **waiting** events and **inactive** processes.

## How Are Overdue Processes Cancelled?

A user may cancel an overdue process at their discretion, online / real-time using [Overdue Process - Main](#).

The system will automatically cancel an overdue process when the object(s) associated with the overdue process are sufficiently paid. Exactly when the system checks if an overdue process should be cancelled is dependent on your organization's billing and accounting rules. For example, if you practice [open-item accounting](#), you'd want to analyze an account's active overdue processes whenever a match event is added, changed or deleted (as match events are the only objects that impact if debt is considered paid in an open-item world). The base-product supports this specific example. Alternatively, if you plug in the base product Account Type - FT Freeze algorithm ([C1-CFTZ-CMRV](#)), an account's overdue processes will be reviewed for cancellation whenever a credit FT is frozen for the account. If you need additional events to check if an overdue process should be canceled, a base-product change MAY be necessary. Please check with customer support if you have questions.

Two algorithms plugged-in on the [overdue process template](#) handle the cancellation:

- The **Cancel Criteria** algorithm is responsible for determining if an overdue process should be canceled. Algorithms of this type analyze the outstanding debt on the objects linked to the overdue process and indicate whether a process can be cancelled.
- The **Cancel Logic** algorithm is responsible for actually canceling the process. The logic involved in cancellation can be quite sophisticated as canceling an overdue process can result in the cancellation of its pending events.

**Why two algorithms?** The reason two algorithms are involved in cancellation is that we want the cancellation logic to be encapsulated in one place so it can be called during both manual and automated cancellation.

**Different logic for different templates.** Because both the *Cancel Criteria* and *Cancel Logic* algorithms are plugged-in on the overdue process's template, you can have different cancellation criteria and logic for different templates.

## Overdue Processes Are Created From Templates

As described above, you set up [overdue process templates](#) to define the types of events and when they are executed. When an overdue process is created, its events are created by copying the event types from an overdue process template. The remaining topics in this section provide background information to assist you in setting up your templates.

## The Big Picture Of Overdue Events

This section describes the various types of overdue events and their lifecycle.

### Contents

- [How Are Overdue Events Created?](#)
- [Overdue Events Can Do Many Things](#)
- [Overdue Event Information Is Overridable](#)
- [Overdue Event Lifecycle](#)

### How Are Overdue Events Created?

Overdue events are created as follows:

- The [Overdue Monitor](#) invokes *Overdue Monitor Rules* to periodically check your accounts (refer to [Overdue Rules Are Embodied In Algorithms](#) for how this works). An *Overdue Monitor Rule* creates an overdue process when an account violates your overdue rules. The overdue process has one or more overdue event(s). The number and type of events is controlled by the overdue process template specified on the *Overdue Monitor Rule*.
- Users can create an overdue process manually on [Overdue Process - Main](#). To do this, they specify an overdue process template. The number and type of overdue events is defaulted from the template.
- An overdue event may be manually added to an existing overdue process by a user on [Overdue Process - Events](#).

**Bottom line.** Most overdue events are created by the system when it creates an overdue process for delinquent debt. If you need to create an ad hoc overdue event, you can either create an overdue process whose template contains the desired event OR link the desired event to an existing process.

### Overdue Events Can Do Many Things

An overdue event can perform a wide number of activities as the logic is embodied in an algorithm. The following points describe how this works:

- Every overdue event references an [overdue event type](#).
- The overdue event type, in turn, references an *Event Activation* algorithm.

- The **Event Activation** algorithm is invoked when the event is [triggered](#).

### Overdue Event Information Is Overridable

“Overdue event info” is the concatenated string of information that summarizes an overdue event throughout the system. The base-product logic constructs this string by concatenating the following information:

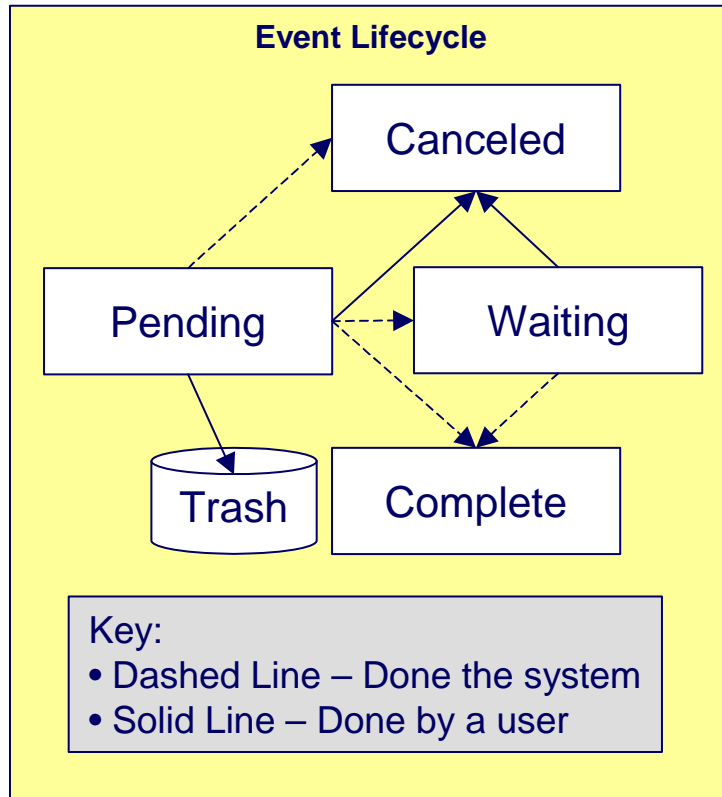
- The event type's description
- The event's status
- If it's **pending**:
  - If the event has a trigger date, the number of days until it's triggered plus the verbiage **day(s) from today**
  - Otherwise, the verbiage **dependent on other events**
- If it's **waiting**, the number of days, hours and minutes that it's been waiting
- If it's **canceled**, the cancel reason code's description
- If it's **complete**, the number of days, hours and minutes that it's been complete

If you'd prefer a different info string, you can develop a new algorithm and plug-it in on your event types. This design allows some / all event types to have an override info string.

### Overdue Event Lifecycle

The following diagram shows the possible lifecycle of an overdue event:





Overdue events are initially created in the **Pending** state. An event can take myriad paths after it's created; it all depends on how you've configured the system. The following topics describe an event's lifecycle:

### Contents

- [How and When Events Are Activated](#)
- [Activating Events Should Add A Log Entry](#)
- [Holding Events](#)
- [Some Events Wait For Something Before They Activate](#)
- [How Are Events Canceled](#)

### How and When Events Are Activated

An overdue event contains the date it should be activated; this is referred to as its trigger date. On this date, the Overdue Event Manager (a background process (**C1-ODET**)) invokes the **Event Activation** algorithm plugged-in on the event's event type. The **Event Activation** algorithm, in turn, will decide on the state in which to leave the overdue event (e.g., it may transition it to the **Complete** state or the **Waiting** state).

If a user can't wait for the Overdue Event Manager real-time, they can click a button on [Overdue Process - Main](#) to activate (and recursively trigger) events online / real-time.

You control how an event's trigger date is populated by configuring the [overdue process template](#). You are given two choices when you link an event type to an overdue process template:

- You can indicate the event should be assigned a trigger date when it is first created. You'd use this approach on the first event and events with no dependencies on earlier events. The following points describe how to configure the overdue process template to do this:

- Indicate the event type is NOT dependent on other events, and
- Define the number of days after the process's creation to use when calculating the trigger date.
- You can indicate the event should be assigned a trigger date only after earlier events are **Complete**. This technique should be used whenever you have an event that is only executed after other events are **Complete**. The following points describe how to configure the overdue process template to do this:
  - Indicate the event is dependent on other events, and
  - Define the number of days after the completion / cancellation of all dependent event(s) that the trigger date should be set to. The Overdue Event Manager sets the trigger date on such an event when it detects that all of its dependent events are complete / canceled.

**Calendar vs. Workdays.** When an overdue event is created by the system, its trigger date is set in accordance with your date arithmetic preferences. Refer to [Calendar vs. Work Days](#) for more information.

### Activating Events Should Add A Log Entry

As described [above](#), an overdue process has a log holding a history of meaningful events in the process's life. Most **Event Activation** algorithms will add an entry to the process's log.

These log entries are more than just an audit trail as they also reference the objects that are created during activation. For example, if an activation algorithm creates a customer contact, the ID of the customer contact will be referenced on the log (and end-users will be able to drill down on the log entry to see the customer contact).

### Holding Events

You can prevent a **pending** event with a trigger date on / before the current date from activating by plugging-in a **Hold Event Activation** plug-in on the overdue process template. This might prove useful, for example, if you want to suspend an overdue process while another process, such as an appeal by the taxpayer, is outstanding. Then, when the other process is complete, the overdue process can start up where it left off. Currently the base provides an algorithm type (**C1-HIOCE**) to suspend an overdue process while an account is linked to an open [case](#) of a given type.

### Some Events Wait For Something Before They Activate

Consider this scenario - you want an overdue event to create a To Do entry so a user can authorize the next phase of an overdue process. When this event activates, the event's activation algorithm will create a To Do entry, but it will NOT transition the event to **complete**. Rather, the overdue event will exist in the **waiting** state. While in the **waiting** state, the Overdue Event Manager will monitor the state of the To Do entry. When the To Do entry completes, the original overdue event can transition to the **complete** state and then latter dependent events can be triggered. The following points describe how to configure the system to support this type of event:

- The event type's **Event Activation** algorithm should behave as follows:
  - It creates the object on which the overdue event waits.

- It must link this object to the overdue process by creating a log entry where the prime-key of the related object is referenced (in a foreign-key characteristic). This log entry should also reference the event.
- It should leave the overdue event in the **waiting** state.
- The event type must have a **Monitor Waiting Event** algorithm. This algorithm is invoked each time the [Overdue Event Manager](#) executes. If the related object has transitioned to a "final" state, the originating overdue event is transitioned to the **complete** state (and then latter dependent events are triggered).

**Bottom line.** Two algorithms must be set up on an overdue event type to implement waiting functionality: an Event Activation algorithm that creates the monitored object and a **Monitor Waiting Event** algorithm to check on the state of the monitored object. The Overdue Event Manager has the dual responsibility of activating the event and monitoring its related object for completion (and then triggering the dependent events when it completes).

While the above example illustrated how an overdue event could create and then monitor a To Do entry, you can use this functionality to create and monitor any object that has an initial and final state. If the base product does not contain the algorithms you need, simply develop new ones using the base-product algorithms as examples.

#### How Are Events Canceled

A **pending** event will be **cancelled** automatically by the system when the overdue process is canceled. Refer to [How Are Overdue Processes Cancelled](#) for more information.

A user may cancel a **pending** or **waiting** event at their discretion.

Regardless of what triggers the cancellation, the **Cancel Logic** algorithm plugged in on the overdue event type handles the cancellation. This allows you to introduce additional cancellation logic should the need arise. Please note that the base product cancel algorithms insert a [log entry](#) when a user manually cancels an event.

## The Big Picture of Collection Cases

In many tax authorities collecting unpaid debt is typically done in two phases: a series of unmonitored actions, typically letters, attempting to convince the taxpayer to pay, and a series of user-oriented actions such as contacting the taxpayer, setting up payment plans, and referring debt to a collection agency. The user-oriented actions are handled using a collection case. While not required, it is expected that many overdue processes will create a collection case when the overdue events did not prompt payment of the debt.

The topics in this section provide background information about collection cases.

#### Contents

- [Collection Case Overview](#)
- [How Are Collection Cases Created?](#)
- [Collection Case Lifecycle](#)
- [How Are Collection Cases Closed?](#)
- [Overdue Events Wait For The Collection Case To Conclude](#)
- [Collection Case Type Defines Parameters](#)

## Collection Case Overview

A collection case provides the functionality to record and track the interactions between the taxpayer and the user responsible for the collection. Many actions can take place once a collection case is created:

- A pay plan can be created
- The debt may be referred to a collection agency
- Letters may be sent to the taxpayer to advise of additional penalties
- A lien may be placed to secure the debt

## How Are Collection Cases Created?

The activation of an overdue event creates a collection case. It is possible for an overdue process to be linked to more than one collection case over time but only one collection case can be active for an overdue process at a given time. The base package provides a sample overdue event activation algorithm ([C1-CR-COL-CS](#)) to create a standard collection case.

Collection cases are created for persons not accounts. It is possible for collection cases that are linked to different overdue processes to be consolidated into a single case.

It is important to note that a collection case exists with respect to one or more overdue processes. A collection case's overdue process defines the objects in arrears. These objects are monitored to determine if the overdue process can be cancelled and the collection case can be closed.

## Collection Case Lifecycle

The lifecycle of the collection case depends upon the configuration of the associated business object. The base package includes a sample standard collection case with a simple lifecycle, as follows:

- The collection case is initially created in the **Pending Investigation** state.
- The case will transition to the **Actions In Progress** state the first time a user initiates an action, such as creating a pay plan or sending a letter. Since many collection case actions happen in parallel, the expectation is that the case remains in this state until closed. Monitoring algorithms can be configured to check whether any of the actions in progress for the case have been waiting too long.
- The case may be **Transferred** to another collection case type or to an existing collection case.
- The user can manually transition the collection case to the **Uncollectible** state if they determine that no further action to collect the debt is possible.

In addition to the standard actions for edit and state transition, a collection case may have special actions that apply to this type of collection, such as creating a pay plan or a collection agency referral. Additional actions are configured via the maintenance UI map. Refer to the base business objects for further details of the sample configuration.

## How Are Collection Cases Closed?

Collection cases can be closed at the user's discretion or when the associated overdue process is cancelled. The base package provides a sample **Cancel Logic** algorithm ([C1-CL-COLLCS](#)) that closes any open collection cases linked to the process provided there are no other active overdue processes for the case.

## Overdue Events Wait For The Collection Case To Conclude

Overdue events that create collection cases are perfect examples of events that [wait](#) for the object they create to complete before they, in turn, **complete**. After the collection case concludes, the originating overdue event will complete thus triggering its dependent events, such as writing off the debt. The base package provides a sample **Monitor Waiting Event** algorithm that checks whether all collection cases associated with the overdue process are in a final state.

## Collection Case Type Defines Parameters

For each type of collection case, you must configure a collection case type to capture the appropriate parameters needed by the collection case actions. Typical parameters include:

- The To Do Type used to notify the responsible user that a new collection case has been created
- The default pay plan type to be used for a collection case of this type
- The To Do Types to be used to notify the responsible user if certain actions have been outstanding for too long without a response, such as a collection agency referral with no updates.

## The Big Picture Of Collection Agency Referrals

Before debt is written off, many implementations refer the unpaid debt to a collection agency. The following topics describe how collection agency referrals are managed.

### Contents

- [Collection Agency Referrals Overview](#)
- [How Are Collection Agency Referrals Created?](#)
- [Cancelling Collection Agency Referrals](#)

## Collection Agency Referrals Overview

The system creates a [Collection Referral](#) record for a collection agency to track the debt that is to be collected. A collection referral is linked to an account. Collection referrals have history records that contain the amount of debt referred to the agency. Creating a history record triggers the interface of information to the collection agency. The method used to interface the information to the agency is defined on the collection agency's record. Refer to [Setting Up Collection Agencies](#) for more information.

## How Are Collection Agency Referrals Created?

Users can create collection agency referral manually or by configuring an overdue event type with an activation algorithm that creates the referral. The base package sample activation algorithm ([C1-OE-AGYREF](#)) will refer the total unpaid debt for the overdue process to the collection agency with the least amount of referred debt. If you prefer different logic, you must develop your own algorithm.

In many tax authorities, creating a collection agency referral is one of the actions that would typically take place for a collection case. The base product BO for collection case **C1-StandardCollectioncase** can be used as an example of how to include the functionality to create a collection agency referral and link it to the collection case.

## Cancelling Collection Agency Referrals

Collection agencies are notified of the cancellation of a referral by the creation of a new collection agency referral history record (with a type of cancel). This record will be interfaced to the agency in the same manner used to interface a new referral (see above). You can cancel a referral manually by simply creating a new collection agency referral history record (with a type of **cancel**).

If the collection agency is successful in obtaining the funds, a payment will be added. If the payment satisfies the cancel criteria defined on the overdue process template's cancellation plug-in, the overdue process will cancel. When an overdue process is cancelled, the cancel criteria on the overdue process's template are executed. If your implementation chooses to create collection agency referrals via overdue events, we strongly recommend plugging in an algorithm that will cancel the referrals when an overdue process is cancelled. If you choose to manage collection agency referrals via collection cases, the base package provides a sample business object status **Enter** algorithm ([C1-CCC-AGCYR](#)) to cancel any referrals linked to the collection case when it is closed.

If the collection agency is not successful in obtaining your funds after a given amount of time, you probably want to cancel the referral and write-off the debt. If your implementation chooses to create collection agency referrals via overdue events, you set can set up your overdue process template to have an event that creates a collection agency cancellation X days after the referral. The base package provides a sample event activation algorithm ([C1-OE-AGYCAN](#)) to cancel all active collection agency referrals associated with the overdue process

**Log entry.** The base-product overdue event activation algorithms that make and cancel collection agency referrals insert rows in the overdue process's [log](#) to audit these events.

## Write Offs Are Implemented Using Overdue Events

The system has been designed to allow overdue events on the original overdue process to write-off the objects being collected.

### Contents

- [Starting Write-Off Oriented Events](#)
- [Small Amount Write-Downs](#)
- [Write Offs And Overdue Process Cancellation](#)

### Starting Write-Off Oriented Events

While the system provides overdue event activation algorithms that manage write-off oriented actions, such as referral to collection agency, tax authorities typically handle those actions as part of collection case activity.

Most overdue process templates will be configured to contain an event that writes-off the unpaid balance of the debt. This event should be configured to be dependent on the event that created the associated collection case. A **Monitor Waiting Event** algorithm can detect the collection case is closed and complete the waiting event. This allows the subsequent write off event to be triggered.

## Small Amount Write-Downs

Many organizations will write-down a debt whose value is small early in an overdue process. The base package overdue event activation algorithm to write off assessments ([C1-WRITE-OFF](#)) includes a parameter to support this requirement. (For example, indicate that you should write down an overdue process's assessment if their value is less than a threshold amount).

If your organization writes-down small amounts differently than large amount, simply set up an overdue event type to reference such an activation algorithm and position it in the appropriate place in the overdue process template.

## Write Offs And Overdue Process Cancellation

If an overdue event writes off debt, the state of the process depends on your cancel criteria and where the overdue event is positioned in the overdue process. For example, if an overdue process has an overdue event that writes off small amounts of debt early in the process, a process whose debt meets the threshold criterion will be canceled when the event activates.

Contrast this to an overdue process where the last event writes off the debt. Because there are no other events to activate, the process will complete (i.e., it will not be canceled).

## Calendar vs. Work Days

When you set up your overdue templates, you supply information that controls how the event's trigger date is calculated. You have two options:

- You can say that an event's trigger date can only be populated after earlier, dependent events are complete. For example, the 2<sup>nd</sup> event is triggered 2 days after the 1<sup>st</sup> event is complete.
- You can say that an event's trigger date is populated when the process is first created. You simply define the number of days after the start of the process when each such event should be triggered. For example, the 2<sup>nd</sup> event can be triggered 7 days after the start of the process.

In addition to the above, an option defined on the [Feature Configuration for Overdue Processing](#) plays a part in the calculation of an event's trigger date:

- If you set the option to use **calendar days**, the trigger date of events will be set to the first workday on / after the calculated date. For example, if you indicate that the 2<sup>nd</sup> event is triggered 7 days after the 1<sup>st</sup> event, the system will add 7 days to the 1<sup>st</sup> event's completion date. It then checks if this is a workday (and not a holiday); if so, this is the trigger date of the event; if not, it assigns the trigger date to the next workday.
- If you set the option to use **workdays**, the trigger date will be calculated by counting workdays. For example, if you indicate that the 2<sup>nd</sup> event is triggered 7 days after the 1<sup>st</sup> event, the system will count 7 workdays and set the trigger date accordingly.

**Account's division controls the work calendar.** The system uses the above information in conjunction with the [work calendar](#) associated with the account's division to determine workdays.



## Creating Overdue Procedures

Your overdue procedures define how your organization collects overdue debt. In this section, we describe how to set up the data that controls these procedures.

**Warning!** There are numerous ways to design your overdue procedures. Some designs will result in easy long-term maintenance; others will result in maintenance headaches. In this section, we provide information to help you understand the ramifications of the various options. Before you set up your overdue procedures, we encourage you to gain an intuitive understanding of these options by using the system to prototype the alternatives.

### Contents

- [Set Up Tasks](#)
- [Setting Up Overdue Processing](#)

## Set Up Tasks

The above topics provided background information about how overdue processing works. The following discussion summarizes the various set up tasks.

### Contents

- [Overdue Event Types](#)
- [Overdue Process Templates](#)
- [Collection Classes](#)
- [Collection Class Overdue Monitor Rules](#)
- [Collections - Installation Options](#)
- [Standard Collection Case Type](#)
- [Feature Configuration](#)
- [Overdue Cancellation Reasons](#)
- [Collection Agencies](#)
- [Alert To Highlight Active Overdue Processes](#)
- [Alert To Highlight Active Collection Cases](#)
- [Bill-Oriented Collection - Additional Set Up](#)

## Overdue Event Types

You will find that most of the time spent setting up your overdue event types is spent setting up the objects that are referenced on the overdue event type algorithms. For example, if you use the base-product algorithms, you will set up the following:

- The various "types" for the objects created by the plug-ins. For example,
  - If an overdue event type creates a To Do entry, you must set up the To Do type.
  - If an overdue event type creates a customer contact, you must set up the customer contact type.
- [Foreign key characteristic types](#) that are used to reference the ancillary objects in the [log entries](#) (e.g., if an event creates a customer contact, the log references this customer contact using a FK characteristic type). Note, many of these will exist in the base-product.



- [Messages](#) that are used to define the verbiage in the [log entries](#). For example, if you use the base-product algorithm that creates a customer contact, you must supply the desired message category and number that contains the verbiage that appears in the log when customer contacts are created. Note, messages have been set up for all base-product algorithms (this means you should not have to set up new messages).

The only way to compile the complete list is to design the parameters for each overdue event type algorithm. Refer to [Overdue Event Type - Main](#) for the supported plug-in spots.

After you've set up the objects referenced on the algorithms, you can then set up the algorithms. Only then can you set up the overdue event types.

## Overdue Process Templates

After your overdue event types exist, you can set up your overdue process templates. You will find that most of the time spent setting up your overdue process templates is spent setting up the objects that are referenced on the overdue process template algorithms. Refer to [Overdue Process Template - Main](#) for the supported system events.

## Collection Classes

When setting up [collection classes](#), make sure to indicate that these collection classes use the **Overdue** collection method (only accounts linked to collection classes designated as using the **Overdue** collection method or processed by the Overdue Monitor).

## Collection Class Overdue Monitor Rules

After your overdue process templates exist, you can set up your **Overdue Monitor Rules**. These rules are algorithms plugged in on [Collection Class Overdue Rules](#). You will find most of the time spent setting up these algorithms is spent setting up the objects referenced on the base-product algorithm.

## Collections - Installation Options

Control tables required to collect on overdue bills are described in [Overdue Processing - Setup Tasks](#). For more information about collections installation options, refer to [Installation Options - Collections](#).

## Standard Collection Case Type

The base product supplies the business object **C1-StandardCollectionCase**, which is designed to cater for collection cases that incorporate standard actions such as creating pay plans and collection agency referrals.

The base product also supplies the business object **C1-StandardCollectionCaseType**, which defines the configuration information used by the standard collection case. To enable this functionality the following configuration tasks are needed:

- Define an appropriate To Do type and To Do role for sending a notification that a new collection case has been created for an overdue process.
- Define a Customer Contact class and type that reference the default letter template to be sent to a taxpayer for a collection case of this type.
- Define an appropriate To Do type and To Do role for sending a notification that the maximum days to wait after sending a letter before additional activity should occur has been exceeded.

- Define an appropriate To Do type and To Do role for sending a notification that the maximum days to wait after referring debt to a collection agency has been exceeded.
- Define a default Pay Plan obligation type for collection cases of this type
- Define appropriate reasons for closing a collection case due to the debt being uncollectible. Uncollectible reasons are defined using a customizable [lookup](#). The lookup field name is **C1\_COLL\_CASE\_UNCOLL\_RSN\_FLG**.
- Define a collection case type for the business object **C1-StandardCollectionCase**

Your implementation can define additional business objects to support collection case processing.

## Feature Configuration

You must set up a [Feature Configuration](#) to define parameters that control various overdue processing options. The following is an example of how the Feature Configuration would look for an implementation:

The following points describe the various **Option Types** that must be defined:

- **Trigger Date: Y-Workdays, N-Calendar Days.** This option controls how the system computes the trigger dates on overdue events. Enter **Y** if the system should use workdays. Enter **N** if the system should use calendar days. Refer to [Calendar vs Work Days](#) for the details.
- **Champion Template\$Challenger Template\$Percentage(1-100).** You need only set up options of this type if your implementation implements [Champion / Challenger](#) functionality. Options of this type are entered in the format **A\$B\$nnn** where A is the overdue process template of the champion template, B is the overdue process template of the challenger template, and C is the percent of the time that the system should create the challenger template. The overdue monitor rule algorithm uses this option to override the champion overdue process template X% of the time with the challenger template. You may enter any number of these options (but only one per Champion Template).

## Overdue Cancellation Reasons

Overdue events can be cancelled automatically and manually (at the discretion of a user). Regardless of the method of cancellation, a cancellation reason must be supplied. You set up your overdue event cancellation reasons using [Overdue Event Cancellation Reason - Main](#).

## Collection Agencies

If you refer debt to collection agents, you must set up your [collection agencies](#).

## Alert To Highlight Active Overdue Processes

If you want an alert to appear if the account has active overdue processes, you must configure an appropriate **Control Central Alert** algorithm ([C1-OD-PROC](#)). This algorithm is plugged in on the [Installation](#) record.

## Alert To Highlight Active Collection Cases

If you want an alert to appear if the account has active collection cases, you must configure an appropriate **Control Central Alert** algorithm ([C1-CCAL-COCS](#)). This algorithm is plugged in on the [Installation](#) record.

## Bill-Oriented Collection - Additional Set Up

The topics in this section provide information on additional set up requirements if you collect on unpaid bills.

### Contents

- [One Bill Per Match Event](#)
- [Alert To Highlight Written Off Bills](#)
- [Open-Item Bill Amount Plug-In](#)

#### One Bill Per Match Event

A bill is considered paid if its financial transactions (FTs) are linked to a **balanced** match event. To determine a bill's outstanding amount, FTs from different bills cannot be commingled on the same match event (but it's OK for a bill's FTs to be on multiple match events). If you stick by the rule of "just one bill per match event", you will then be able to determine the outstanding balance of a partially paid bill. However, if you mix more than one bill under a match event, a particular bill's balance may become indeterminate.

The following Open-Item algorithm types have been provided by the base product to help enforce this rule:

- The Distribute by Bill Due Date **Payment Distribution** algorithm ([C1-PYDS-BDU](#)).
- The match by Bill ID **Payment Distribution Override** algorithm ([C1-PDOV-PYBL](#)).
- The FT cancellation **FT Freeze** algorithm ([C1-CFTZ-COFT](#)).

If any of your customized plug-ins and processes create match events, it is important that these too enforce this rule. You may want to refer to the base product algorithms as an example of how to do this.

#### Alert To Highlight Written Off Bills

If you want an alert to appear if the account has bills with written-off debt, you must configure an appropriate **Control Central Alert** algorithm ([C1-WO-BILL](#)). This algorithm is plugged in on the [Installation](#) record.

#### Open-Item Bill Amount Plug-In

You must set up the algorithm that computes the original, unpaid, and write-off amounts of your open-item bills. This algorithm is called by other algorithms when these amounts are needed. This algorithm is plugged-in on [Installation](#) in the **Determine Open Item Bill Amounts** spot.

## Setting Up Overdue Processing

The topics in this section describe how to set up the control tables to implement your overdue processing and collection procedures.

### Contents

- [Setting Up Overdue Event Types](#)
- [Setting Up Overdue Process Templates](#)
- [Setting Up Collection Classes](#)
- [Setting Up Collection Class Overdue Rules](#)
- [Setting Up Overdue Event Cancellation Reasons](#)
- [Setting Up Collection Case Types](#)
- [Setting Up Collection Agencies](#)

## Setting Up Overdue Event Types

An overdue event type encapsulates the business rules that govern a given type of overdue event. Open this page by selecting **Admin Menu, Overdue Event Type**.

**Recommendation.** Before using this transaction, we strongly recommend that you review [The Big Picture Of Overdue Events](#).

### Description of Page

Enter a unique **Overdue Event Type** code and **Description** for the overdue event type.

Use **Long Description** to provide a more detailed explanation of the purpose of the overdue event type.

The **Algorithms** grid contains algorithms that control important functions. You must define the following for each algorithm:

- Specify the algorithm's **System Event** (see the following table for a description of all possible events).
- Specify the **Algorithm** to be executed when the System Event executes. Set the **Sequence** to **10** unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event** (note, all system event's are optional and you can define an unlimited number of algorithms for each event).

System Event	Optional / Required	Description
<i>Cancel Logic</i>	Required	This algorithm is executed to cancel an overdue event. Refer to <a href="#">How Are Events Canceled</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Event Activation</i>	Required	This algorithm is executed to activate an overdue event on its trigger date. Refer to <a href="#">Overdue Events Can Do Many Things</a> and <a href="#">How and When Events Are Activated</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Event Information</i>	Optional - only used if you want to override an overdue event's info string	This algorithm is executed to construct an overdue event's override info string. Refer to <a href="#">Overdue Event Information Is Overridable</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Monitor Waiting Events</i>	Optional - only used if events of this type can enter the <i>Waiting</i> state	This algorithm is invoked by the Overdue Event Manager for events in the Waiting state. Refer to <a href="#">Some Events Wait For Something Before Completing</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_OD\\_EVT\\_TYPE](#).

## Setting Up Overdue Process Templates

An overdue process template encapsulates the business rules that govern a given type of overdue process. Open this page by selecting **Admin, Overdue Process Template**.

**Recommendation.** Before using this transaction, we strongly recommend that you review [The Big Picture Of Overdue Processes](#).

### Description of Page

Enter a unique **Overdue Process Template** and **Description** for the overdue process template.

**Collecting On Object** defines the type of object managed by this overdue process. This field actually references a [foreign key characteristic type](#) that references the managed object. For example, if this overdue process template manages overdue bills, you'd reference a foreign key characteristic that references the bill object.

The **Algorithms** grid contains algorithms that control important functions. You must define the following for each algorithm:

- Specify the algorithm's **System Event** (see the following table for a description of all possible events).
- Specify the **Algorithm** to be executed when the System Event executes. Set the **Sequence** to **10** unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event** (note, all system event's are optional and you can define an unlimited number of algorithms for each event).

System Event	Optional / Required	Description
<i>Calculate Unpaid &amp; Original Amount</i>	Required	This algorithm is executed to calculate the unpaid and original amounts of the objects associated with the overdue process. These amounts are shown on the overdue process page and in the base-product <a href="#">overdue info string</a> .  Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Cancel Criteria</i>	Required	This algorithm is executed to determine if an overdue process can be cancelled. Refer to <a href="#">How Are Overdue Processes Cancelled</a> for the details.  Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Cancel Logic</i>	Required	This algorithm is executed to cancel an overdue process. Refer to <a href="#">How Are Overdue Processes Cancelled</a> for the details.  Click <a href="#">here</a> to see the algorithm types available for this system event.
<i>Hold Event</i>	Optional - only used if	This algorithm is executed to determine if the activation of overdue

<b>Activation Criteria</b>	overdue processes of this type can be suspended while some condition is true	events should be suspended. Refer to <a href="#">Holding Events</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.
<b>Overdue Process Information</b>	Optional - only used if you want to override an overdue process's info string	This algorithm is executed to construct an overdue process's override info string. Refer to <a href="#">Overdue Process Information Is Overridable</a> for the details. Click <a href="#">here</a> to see the algorithm types available for this system event.

The **Event Types** control the number and type of overdue events linked to an overdue process when it is first created. The information in the scroll defines these events and the date on which they will be triggered. The following fields are required for each event type:

- **Event Sequence.** Sequence controls the order in which the overdue event types appear in the scroll.
- **Overdue Event Type.** Specify the type of overdue event to be created.
- **Days After.** If **Dep on Other Events** is on, events will be triggered this many days after the completion of the dependent events (specified in the grid). Set this value to 0 (zero) if you want the event triggered immediately after the completion of the dependent events. If **Dep on Other Events** is off, events will be triggered this many days after the creation of the overdue process. Refer to [How and When Events Are Activated](#) for the details.
- If **Dep on Other Events** is on, define the events that must be completed or cancelled before the event will be triggered.
  - **Sequence** is system-assigned and cannot be specified or changed.
  - **Dependent on Sequence** is the sequence of the dependent event.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_OD\\_PROC\\_TMP](#).

### Setting Up Collection Classes

Every account has a collection class. This class is one of several fields that control the collection method applied to the account's debt. Open **Admin Menu, Collection Class** to define your collection classes.

#### Description of Page

Enter a unique **Collection Class** code and **Description** for each collection class.

Indicate a **Collection Method** of **Overdue** if the accounts belonging to this collection class are subject to overdue collection procedures. If accounts belonging to this collection class are not subject collection, select **Not Eligible For Collection**. Please be aware that these accounts will NOT be reviewed for overdue debt.

#### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_COLL\\_CL](#).

## Setting Up Collection Class Overdue Rules

Collection class overdue rules contain algorithm that impact accounts associated with a given collection class, division and currency code are managed. Open this page by selecting **Admin, Collection Class Overdue Rules**.

**Recommendation.** Before using this transaction, we strongly recommend that you review [Different Overdue Rules For Different Taxpayers](#).

### Description of Page

Enter the **Collection Class**, **Division** and **Currency Code** to which the rules apply.

The **Algorithms** grid contains algorithms that control important functions. You must define the following for each algorithm:

- Specify the algorithm's **System Event** (see the following table for a description of all possible events).
- Specify the **Algorithm** to be executed when the System Event executes. Set the **Sequence** to **10** unless you have a **System Event** that has multiple **Algorithms**. In this case, you need to tell the system the **Sequence** in which they should execute.

The following table describes each **System Event**.

System Event	Optional / Required	Description
<i>Overdue Monitor Rule</i>	Required	<p>This algorithm is invoked by the Overdue Monitor to analyze an account's debt. Refer to <a href="#">How Does The Overdue Monitor Work</a> for the details.</p> <p>If you have multiple rules (and therefore multiple algorithms), please take care when assigning the sequence number, as the Overdue Monitor will invoke these rules in sequence order.</p> <p>Click <a href="#">here</a> to see the algorithm types available for this system event.</p>

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_OD\\_RULE\\_ALG](#).

## Setting Up Overdue Event Cancellation Reasons

An overdue event cancel reason must be supplied before an overdue event can be canceled. Open this page by selecting **Admin, Overdue Event Cancel Reason**.

### Description of Page

Enter an easily recognizable **Overdue Event Cancel Reason** and **Description** for each cancellation reason.

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_OEVT\\_CAN\\_RSN](#).



## Setting Up Collection Case Types

A **Collection Case Type** defines the configuration information that is common to collection cases of a given type. The type of information captured on the collection case type is governed by the collection case type's business object.

To set up a Collection Case Type, select **Admin Menu, Collection Case Type**.

The topics in this section describe the base-package zones that appear on the Collection Case Type portal.

### Contents

- [Collection Case Type List](#)
- [Collection Case Type Actions](#)
- [Collection Case Type](#)
- [Collection Case Type Log](#)

### Collection Case Type List

The Collection Case Type List zone lists every collection case type. The following functions are available:

- Click a [broadcast](#) button to open other zones that contain more information about the adjacent collection case type.
- The standard actions of **Edit**, **Duplicate** and **Delete** are available for each collection case type.
- State transition buttons are available to transition the collection case type to an appropriate next state.
- Click the **Add** link in the zone's title bar to add a new collection case type.

### Collection Case Type Actions

This is a standard [actions zone](#). The Edit, Delete and Duplicate actions and appropriate state transition buttons are available.

### Collection Case Type

The Collection Case Type zone contains display-only information about a Collection Case Type. This zone appears when a Collection Case Type has been broadcast from the Collection Case Type List zone or if this portal is opened via a drill down from another page.

Please see the zone's help text for information about this zone's fields.

### Collection Case Type Log

This is a standard [log zone](#).

### Where Used

Follow this link to open the data dictionary where you can view the tables that reference [CI\\_COLL\\_CASE\\_TYPE](#) and [CI\\_COLL\\_CASE](#).

## Setting Up Collection Agencies

You must set up a collection agency for each such organization to which you refer delinquent debt. To define a collection agency, select **Admin Menu, Collection Agency**.



**Description of Page**

Enter an easily recognizable **Collection Agency** code and **Description** for each collection agency.

A collection agency must be associated with a Person. Choose the **Person ID** of the organization from the prompt.

Information about how to set up persons is discussed in [Maintaining Persons](#).

Turn on the **Active** switch if the collection agency is actively receiving referrals.

Specify the **Batch Control** that's used to route new and cancelled referrals to the collection agency. The batch control's description is displayed adjacent.

**Where Used**

Collection agencies are assigned to collection agency referrals when the collection agency referral background process executes. Refer to [The Big Picture Of Collection Agency Referrals](#) for more information.

# Configuring Zones

Many zones in Oracle Enterprise Taxation Management do not require configuration by your implementation team. For example, the base package is shipped with the **Account Financial History** zone that appears on the **Control Central - Account Information** portal. This zone does not require configuration because its zone type has no configurable options (i.e., its behavior is static).

Other zones require configuration before they can be used because their behavior is dynamic. The topics in this section provide tips and techniques on how to configure zones in Oracle Enterprise Taxation Management.

Refer to [The Big Picture of Portals and Zones](#) for a description of portal and zone functionality.

## Configuring Timeline Zones

A timeline zone contains one or more "lines" where each line shows when significant events have occurred. For example, you can set up a timeline zone that has two lines: one that shows when payments have been received from a taxpayer, and another that shows when bills have been sent to the taxpayer.

For a complete description of the numerous features available on a timeline zone, refer to [Timeline Zone - Account Info](#).

The following points describe how to set up a timeline zone:

- Set up an [algorithm](#) for each line in the zone. These algorithms will reference an algorithm type that is plugged into the **Zone - Timeline Line** plug-in spot. Click [here](#) to see the algorithm types available for this plug-in spot. Please note the following about the parameter values defined on these algorithms:
  - You can set up a timeline algorithm to show an object's "info string" when a user clicks on an event on a timeline. The object's info string appears in the zone's info area. When a user clicks on an "info string", they are transferred to a page (typically the one used to maintain the object). For example, if a user clicks on a "bill info" line, they will be transferred to the bill maintenance page.

You control the format of the info string and the destination transaction by defining the appropriate [foreign key reference](#) in each timeline algorithm's parameters. For example, if you were setting up the algorithm for a bill line, you'd reference the foreign key reference used to show bill foreign keys throughout the system.

- You can set up a timeline algorithm to show [BPA scripts](#) when a user clicks on an event on a timeline. For example, if you click on a bill event, BPA script descriptions can appear in the info area. When a user clicks on one of these descriptions, the script will execute and guide them through a respective business process (e.g., initiate a bill dispute, request a bill reprint, etc.). You define the scripts in each timeline algorithm's parameters.

When a script is initiated from a timeline, the system puts the prime key of the event into a field in the page data model. The name of the field is the column name(s) of the event's prime key. For example, when a script associated with a bill event is kicked off, the system populates a field called BILL\_ID with the prime-key of the selected bill.

The script can use these page data model field to navigate to the pertinent pages. For example, if you were setting up a script to reprint a bill, the first line of the script would reference a navigation option to transfer the user to the Bill - Routing page where they can initiate the reprint. This navigation option will contain context fields that matched the names of the fields in the page data model (this is how field values are passed to pages).

- You can control every color and icon shown on a timeline by specifying the appropriate color codes on the zone's parameters.
- Set up a [zone](#) that references these algorithms. The zone will reference the **F1-TIMELINE** zone type.
- Link the zone to the appropriate portal(s) (e.g., [Control Central - Account Information](#) or [Control Central - Taxpayer Information](#)).
- Update your users' [portal preferences](#) and [security rights](#) so they can see the zone in the desired location on the portal(s).

You can set up many timeline zones. For example,

- You might want different zones to appear on a portal depending on the type of user. For example, you might want one timeline for collection clerks, and a different one for customer service representatives.
- For aesthetic reasons, you might want multiple simple timeline zones to appear on a given portal rather than one complex timeline zone.
- You might want to set up context specific timeline zones. For example, you might want to have one timeline zone that is location-oriented and another that is person-oriented.

# ConfigLab Addendum

This chapter is an addendum to the [general ConfigLab chapter](#). This addendum describes ConfigLab functionality that is specific to Oracle Enterprise Taxation Management.

## Account Staging

---

Some organizations periodically transfer account-related data (e.g., bills, payments, etc.) to a "test" environment so realistic data can be used to test new configuration data. If your organization does this, you can use this transaction to define the "stable" of accounts to be periodically transferred. Use **Admin, Account Staging** to access this transaction.

### Description of Page

This page is dedicated to a grid that shows the accounts to be copied to an environment by a compare DB process. These accounts are defined in respect of a given Environment Reference (meaning you can have a different set of accounts for different target environments). The following information appears in the grid.

- **Account ID** identifies a given account.
- **User** defines the user who added the account.
- **Create Date/Time** is the date and time the account was added to account staging.

**Update the DB process that manages the transfer.** After defining these accounts, you should update the DB process that transfers accounts to have a primary instruction that references the **ACCT STG** maintenance object. You do this in the environment to which the above accounts will be transferred (as this is the environment that you'll submit the comparison process in).

# CTI-IVR Integration

Oracle Enterprise Taxation Management provides tools to facilitate the integration with your Computer Telephony Integration/Interactive Voice Response (CTI/IVR) system. The interface provides the following functionality:

- The ability to launch Control Central for a particular account ID or phone number from an external application
- The ability to perform an outbound phone call from within Oracle Enterprise Taxation Management
- The ability to accept the next call, as dictated by the CTI software, from the toolbar

This document provides technical information needed by your implementers to fully integrate with your CTI/IVR system.

## Contents

[Launching The System From an External Application](#)  
[Initiating an External Call](#)  
[Receiving the Next Caller in the Queue](#)  
[ActiveX Component - CDxCTI](#)

## Launching The System From an External Application

---

The following sections describe possible options to launch the system from an external system.

## Contents

[Launching Control Central Using an ActiveX Navigator](#)  
[Launching The Application Using a URL](#)

## Launching Control Central Using an ActiveX Navigator

Oracle Enterprise Taxation Management provides an ActiveX component **CDxCTI.CDxNavigator** that external applications, such as an IVR application, can use to launch Control Central for a given account number or phone number.

**Enable ActiveX.** In order to use the navigator, you must [configure your browser to enable ActiveX](#).

The CDxNavigator object exposes two methods:

- **ControlCentralByAccountId** invokes Control Central for a given account ID.
- **ControlCentralByPhone** invokes Control Central for a given phone number.

## Contents

[Method: ControlCentralByAccount](#)  
[Method: ControlCentralByPhone](#)  
[Main Processing](#)

### Navigation Sample Provided with the System

#### Method: ControlCentralByAccount

##### Input:

- **Server URL:** The Oracle Enterprise Taxation Management server URL, for example `http://spl-server`
- **AccountId:** The account ID to search for.

VB Example: Navigate to Control Central Using an Account ID

```
Dim nav As New CDxTAPI.CDxNavigator
nav.ControlCentralByAccountId ("http://spl-server:1000", AccountID)
```

Web Page Example: Navigate to Control Central Using an Account ID

```
Dim nav As New ActiveXObject("CDxTAPI.CDxNavigator");
nav.ControlCentralByAccountId ("http://spl-server:1000", AccountID)
```

#### Method: ControlCentralByPhone

##### Input:

- **Server URL:** The Oracle Enterprise Taxation Management server URL, for example `http://spl-server`
- **PhoneNumber:** The phone number of the person to search for
- **PhoneFormat:** The format in which the phone number is provided

VB Example: Navigate to Control Central Using a Phone Number

```
Dim nav As New CDxTAPI.CDxNavigator
nav.ControlCentralByPhone ("http://spl-server:1000", "(415) 357-5423",
"(999) 999-9999")
```

Web Page Example: Navigate to Control Central Using a Phone Number

```
Dim nav As New ActiveXObject("CDxTAPI.CDxNavigator");
nav.ControlCentralByPhone ("http://spl-server:1000", "(415) 357-5423",
"(999) 999-9999");
```

### Main Processing

The ActiveX component performs the following:

- Locate the first Internet Explorer instance running Oracle Enterprise Taxation Management.
- If an Internet Explorer session is found, use ActiveX automation to navigate to Control Central. Depending on the search type (account or phone), enter the appropriate values in the Control Central page and launch the search.
- If an Internet Explorer session cannot be found, launch Internet Explorer and go directly to Control Central.

### Navigation Sample Provided with the System

An HTML page has been provided to demonstrate the integration. This sample may be found in the following location on your Oracle Enterprise Taxation Management server:  
`/ci/cti/CTISample.HTM`.

- Start an Internet Explorer session.
- Navigate to URL above.
- Select an account ID or phone number and click **Navigate**.

**Sample Data.** The accounts and phone numbers included in the sample HTML file correspond to accounts and phone numbers that exist in the demo database.

- A new session is started if one is not already active and Control Central is launched with the account corresponding to the selected account or phone number.

This sample program is provided to illustrate to implementers how to integrate their CTI-IVR system with Oracle Enterprise Taxation Management.

## Launching The Application Using a URL

You may also launch the application using a URL. With this option you can set the system to launch a script upon startup. You can also indicate to the system to automatically load an appropriate page (if this information is not part of the script).

Refer to [Launching A Script When Starting The System](#) for further information.

## Initiating an External Call

This section describes the automated dialer functionality provided with the system as well as information about integrating with your own automated dialer.

### Contents

- [Overview of Automated Dialer](#)
- [Technical Implementation of Automated Dialer](#)
- [Customize Integration to Your Automated Dialer Software](#)
- [Customize Automated Dialer User Interface](#)

## Overview of Automated Dialer

In order to initiate a call to a taxpayer from within the system, a context menu item **Go To Automated Dialer** is available on the Person context menu. To call a taxpayer displayed in the current context, choose this option from the person context menu and a window appears, showing a list of phone numbers defined for that person.

Select the desired phone number and click **Dial**.

**Context Entry Secured.** The [navigation key](#) for this window **automatedDialer** refers to an application service to facilitate application security. If your installation does not support an integration with external dialer software, configure the security settings to ensure that users do not have access to the application service for this context entry.

## Technical Implementation of Automated Dialer

The popup window is implemented as a JSP page, which calls the JSP page ext\_cti\_dialer.jsp to integrate with an automated dialer. The ext\_cti\_dialer.jsp page provided with the system integrates to the Microsoft Phone Dialer, available on any Windows 2000 workstation.

The Microsoft Phone Dialer is invoked through the CDxPhoneDialer ActiveX object.

```
*****
* Invoke an external phone Dialer
* In the sample provided we launch the Microsoft phone Dialer
*****
*/
function callDialer(phoneNumber){
    CDxPhoneDialer.makeCall(phoneNumber);
}
```

### Object: CDxPhoneDialer

This object is used to call the Microsoft Phone Dialer for an outbound call. It is only used when your implementation uses the Microsoft standard dialer.

### Method: makeCall

**Input:** Phone Number

## Microsoft Phone Dialer Configuration

If your implementation chooses to use the functionality provided with the system and integrate with Microsoft Phone Dialer, you must perform the following steps:

- Copy the JSP page ext\_cti\_dialer.jsp from the /cm\_templates directory found under the web application root directory on your Oracle Enterprise Taxation Management server to the /cm directory.

**Enable ActiveX.** In order to use the integration with the Microsoft Phone Dialer, you must [configure your browser to enable ActiveX](#).

## Customize Integration to Your Automated Dialer Software

In order to integrate with a different automated dialer software application, your implementers must modify the ext\_cti\_dialer.jsp to call the appropriate dialer.

- Copy the JSP page ext\_cti\_dialer.jsp from the /cm\_templates directory found under the web application root directory on your Oracle Enterprise Taxation Management server to the /cm directory.
- Make the appropriate changes to the copy of ext\_cti\_dialer.jsp in the /cm directory to integrate with your automated dialer.



## Customize Automated Dialer User Interface

Your implementation may choose to display a different user interface for the **Go To Automated Dialer** function than the one provided with the system. For example, perhaps there is more information that you would like to display in addition to the person's name and phone numbers. In order to do this, perform the following steps:

- Create your customized component to provide the desired functionality.
- Create a navigation key for your new component and indicate the URL being overridden. The remainder of the section walks you through these steps.

Go to **Admin, Navigation Key +**.

For **Navigation Key**, specify a name for the new navigation key prefixed with CM.

For **URL Location**, select **External (Override)** to override a base navigation key.

When you select **External (Override)**, the **Overridden Navigation Key** becomes available. Select the **automatedDialer** navigation key because that is the key you are overriding.

The **URL Override** is the path on the Web server to your custom component.

When overriding a navigation key, you must flush the system login cache on the Web server. The navigation keys are stored in the system login cache, so the overrides do not become effective until the cache is flushed. To flush the cache, issue the following command in your browser's address bar: **<http://server:port/flushSystemLoginInfo.jsp>**, where server is the name or address of your web server and port is the port number of the application, for example, **<http://CD-Implementation:7500/flushSystemLoginInfo.jsp>**.

Refer to the [Defining Navigation Keys](#) for more information.

## Receiving the Next Caller in the Queue

If your CTI-IVR system allows users to request the next caller waiting in a queue, the system provides a mechanism to integrate with this functionality.

A **Next Call** button is available in the toolbar that can be used to request the next call waiting in an inbound queue managed by a CTI application. This button is only enabled if you have set the **CTI Integration** flag to **Yes** on your [installation options](#).

When the next call button is clicked, it launches a browser script function called **launchCTI** located in a file called **ext\_cti.jsp**. The **launchCTI** function calls a function called **ctiGetNextCaller** to retrieve the next caller's account ID and uses the **CDxNavigator** object to launch Control Central for that account.

## Customize Integration to Your Next Caller Function

The **ext\_cti.jsp** file shipped with the base product provides sample functionality that should be replaced with the appropriate integration to your CTI application. In the sample provided, the **ctiGetNextCaller** randomly takes an account ID from a predefined list of accounts.

In order to integrate the next caller functionality with your CTI-IVR system, perform the following steps:

- Copy the JSP page `ext_cti.jsp` from the `/cm_templates` directory found under the web application root directory on your Oracle Enterprise Taxation Management server to the `/cm` directory.
- In the `/cm` directory, replace the contents of the *`ctiGetNextCaller`* function to retrieve the next caller ID from your CTI application.

## ActiveX Component - CDxCTI

---

The system provides an ActiveX component **CDxCTI** that contains all the functionality required for inbound and outbound calling. It contains two objects:

- CDxNavigator
- CdxPhoneDialer

### Contents

[Configuring Your Browser to Enable ActiveX](#)

[Installing the CDxCTI ActiveX Component](#)

[Creating an Instance of the CDxCTI Object in a Web Page](#)

## Configuring Your Browser to Enable ActiveX

In order to use the automated phone dialer functionality or the next caller functionality, your users must configure their browser to enable ActiveX. Perform the following steps:

- In your Internet Explorer browser window, navigate to **Tools, Internet Options** and go to the **Security** tab. From there, select **Local Intranet**.
- Click **Custom Level**.
- Under the ActiveX controls and plug-ins section, set the following:
  - Download signed ActiveX controls: *Prompt*
  - Download unsigned ActiveX controls: *Disable*
  - Initialize and script ActiveX controls not marked as safe: *Disable*
  - Run ActiveX controls and plug-ins: *Enable*

## Installing the CDxCTI ActiveX Component

The CDxCTI ActiveX components install automatically the first time the Automated Phone Dialer is launched or when the `CTISample.htm` is launched. The CDxCTI component is signed using Microsoft Authenticode technology, therefore when it is downloaded the first time, a dialog will appear describing the source of the component and asking the user to accept the installation of the component on the local machine.

## Creating an Instance of the CDxCTI Object in a Web Page

To use the CDxCTI objects from a web page, declare them explicitly using the `OBJECT` tag:

```
<OBJECT ID="CDxPhoneDialer"  
CLASSID="CLSID:151A6E91-8C55-4666-BFFB-9EC345583CBD"  
CODEBASE="CDxCTI.CAB#version=1,5,0,8">  
</OBJECT>  
  
<OBJECT ID="CDxNavigator"  
CLASSID="CLSID:E7EF882D-662A-4451-A78C-CD62393F06C6"  
CODEBASE="CDxCTI.CAB#version=1,5,0,8">  
</OBJECT>
```

Alternatively, use the new ActiveXObject function

```
var nav = new ActiveXObject("CDxCTI.CDxNavigator");  
  
var nav = new ActiveXObject("CDxCTI.PhoneDialer");
```

# Web Self Service

The Web Self Service (WSS) application is a Web application that is used by a taxpayer of an Oracle Enterprise Taxation Management installation to query and update taxpayer information. WSS provides installations with a framework and templates for implementing transactions typical to a WSS application. In addition some sample web pages are provided.

This section describes the sample pages provided and includes guidelines for an implementer to follow when adding additional WSS pages to the pages provided.

## Contents

- [WSS Pages Provided with the System](#)
- [Customizing Web Self-Service](#)

## WSS Pages Provided with the System

The system provides a limited number of pages for the web self-service application. These pages are provided as an example of how you may implement your WSS application. All the pages provided by the system may be used by your implementers as a sample to use for incorporating such a page into your web application.

Refer to [Customizing Web Self-Service](#) for more information.

## Contents

- [Login Related Pages](#)
- [Account and Person Related Pages](#)

## Login Related Pages

The following pages are related to registering and logging in to the WSS application.

## Contents

- [Login Page](#)
- [Registration Page\(s\)](#)
- [Password Reminder Page](#)

## Login Page

The login page allows a registered user to login to the WSS application with a user name and password.

The page also provides links to enable the user to register as a new user and to request a reminder for a forgotten password.

**Customer Contact Added.** A customer contact is added when a taxpayer successfully logs in or if a taxpayer's attempt to login failed if you have set up the appropriate customer contact classes and types in the configuration properties file. Contact your system administrator for more information.

## Registration Page(s)

The registration pages allow a taxpayer to sign up for web self-service.

- The initial Customer Registration page asks the taxpayer to provide the account number and name in order for the system to identify the taxpayer's person record.
- Once the taxpayer's person record has been identified, the taxpayer is asked to provide their email address and to define a user id and password. The taxpayer must also choose a password reminder question and indicate an answer. This information is stored on the [person](#) record.

Upon successful registration, an email confirmation is sent to the taxpayer.

**Enabling Email Correspondence.** Sending an email is only possible if the WSS application has been configured with the appropriate settings for communicating with an SMTP server. Contact your system administrator to verify that these settings have been defined.

**Preventing Web Access.** A taxpayer may only view information about related accounts if the [account / person](#) record has been marked to allow web access.

## Password Reminder Page

This page is displayed if the taxpayer has clicked the "forgot password" hyperlink from the login page. The password reminder question is displayed and the taxpayer is asked to provide the answer. If the correct answer is provided, an email is sent to the taxpayer's email address with the user name and password.

**Enabling Email Correspondence.** Sending an email is only possible if the WSS application has been configured with the appropriate settings for communicating with an SMTP server. Contact your system administrator to verify that these settings have been defined.

## Account and Person Related Pages

The pages described in this section are available once the taxpayer has successfully logged in.

### Contents

- [Account Summary](#)
- [Account Information](#)
- [View Account Financial History](#)
- [View Billing History](#)
- [Make a Payment](#)
- [Update Personal Info](#)
- [Change Password](#)

## Account Summary

The My Accounts page displays after logging in if the taxpayer's person record is linked to more than one account. For each account, information about the main taxpayer is displayed, along with the account's balance.

Select one of the accounts and you are brought to the Account Information page.

### Account Information

The Account Information page displays high-level information about the person and account. This page also displays a list of obligations linked to the account.

### View Account Financial History

The Account Financial History page displays the financial history for the account. The page is similar to the account financial history page in Oracle Enterprise Taxation Management, with some minor differences. For example, only the current amount and current balance information is displayed. The payoff amount and payoff balance are not displayed. In addition, the taxpayer does not see any adjustment whose adjustment type's print by default flag is not checked.

### View Billing History

The Billing History page displays the information about the taxpayer's bills including the date, amount and the running balance.

### Make a Payment

The Make Payment page allows the taxpayer to enter a credit card payment to pay all or part of the outstanding balance for the account.

**Configuration Required.** When setting up the WSS application, an appropriate tender type and appropriate auto pay source codes should have been defined in the system and referenced in the configuration properties file. Contact your system administrator for more information.

**Customer Contact Added.** A customer contact is added when a taxpayer enters a credit card payment if you have set up the appropriate taxpayer contact class and type in the configuration properties file. Contact your system administrator for more information.

### Update Personal Info

The Update Personal Info page allows a taxpayer to change personal information such as phone numbers, email address and mailing address.

**Customer Contact Added.** A customer contact is added when a taxpayer has changed personal information if you have set up the appropriate customer contact class and type in the configuration properties file. Contact your system administrator for more information.

**To Do Entry Added.** A To Do entry is added when a taxpayer has changed personal information if you have set up the appropriate To Do type in the configuration properties file. Contact your system administrator for more information.

## Change Password

This page allows the taxpayer to change their password. After the password is changed, an email is sent to the taxpayer's email address with the user name and password.

**Enabling Email Correspondence.** Sending an email is only possible if the WSS application has been configured with the appropriate settings for communicating with an SMTP server. Contact your system administrator to verify that these settings have been defined.

## Customizing Web Self-Service

---

WSS provides Oracle Enterprise Taxation Management taxpayers with a framework for implementing transactions. This framework enables the expansion of the functions available to your taxpayer to include other business functions supported by the system. The following sections identify the different possibilities for customizing and expanding the WSS application.

**Upgrading after Customization.** If you would like to modify any of the web application functionality provided by the system, you must make copies of the files provided and only make your changes in the copied files. If you have followed this procedure, every effort will be made to provide a seamless upgrade of the web application for bug fixes and future releases.

### Contents

- [Technical Overview](#)
- [Modifying Page Layout](#)
- [Adding a New Transaction](#)
- [Modifying the Menu](#)
- [Style Sheet Modifications](#)
- [WSS Standards](#)

## Technical Overview

The Self Service application is a regular J2EE Web application, conforming to the Servlet 2.2 specification. It uses JSP for the front end and a Java servlet in the back end. All Oracle Enterprise Taxation Management access is done through XAI and there is no direct connection to the database. The technical design is based on model view controller architecture.

The presentation layer is separated from the data access and business rule layer. The JSP pages contain plain HTML (there are no applets, and very little JavaScript - close to none) with the visual layout of the screens. Dynamic elements, such as taxpayer name, address, balance, etc. are represented as Java Beans (but not EJBs) and are embedded in the page using JSP directives. It is very easy for a Web designer to modify or replace the graphic design of the page without programming knowledge. All the logic of the application, including communication with XAI, sits in the Java Beans (Java classes representing data) and in the servlet.

Each of the Java Beans represents, roughly, an object in Oracle Enterprise Taxation Management, or the set of information returned by an XAI service. Each bean knows how to retrieve its data through XAI if needed, and how to make updates (e.g. enter a payment).

The servlet is the controller that holds the pages together and implements security. Every user request gets intercepted by the servlet, which first checks that the user has been authenticated, performs any background actions if necessary, creates Java Beans with all the information to be displayed to the user, and passes the beans to the appropriate JSP page. Each action in Self Service, for example, show account information, show financial history, make payment, is a separate class.

## Modifying Page Layout

If you would like to keep the functionality provided by a given web application page, but you would like to change the look and feel of the page, perform the following steps:

- Find the JSP file for the page that you want to modify. The JSP files are found in the SelfService directory.
- Copy the file and name your new file the same name as the file you are copying, but with "CM" in the beginning of the file name.
- Make the desired changes in your new JSP file.

Assuming that you want all the other functionality to remain the same, you are finished. You do not need to modify the menu or any action classes to call your new CM JSP file. The functionality provided always looks for the existence of a "CM" version of the JSP file and uses that file instead if one is found. If a "CM" version is not found, the JSP provided by the system is used.

## Adding a New Transaction

As explained in the [technical overview](#), there are two parts for each transaction.

- On the client side, the JSP page contains the UI information
- On the server side, a servlet administers the processing of the transaction using one or more java beans containing the logic to communicate with Oracle Enterprise Taxation Management.

If you want to add another action, for example enroll in AutoPay, you must do the following:

- Write another class for the servlet to recognize the action
- Write another bean that would know how to capture AutoPay information
- Add one or more JSP pages for the new screens that are used to interact with the taxpayer.

**Naming Convention.** In order to ensure that new files added in future releases do not override your files, be sure to use "CM" for "customer modification" for the first two letters of every new file.

### Contents

- Identifying a Transaction
- Creating an XAI Schema
- Creating a Java Bean
- Creating an Action Class
- Control Classes
- Creating a JSP Page



## Identifying a Transaction

In principal, every transaction that exists in the system could get implemented in the WSS, since all Oracle Enterprise Taxation Management objects are described as Tuxedo services, which can get accessed via XAI.

The question often will be whether the implementer will want to maintain the degree of diversity and sophistication that exists in many transactions, or if the maintenance of the main object attributes might already fulfill the requirements.

For reasons of simplicity, the following sections demonstrate the implementation of the service address maintenance, which is handled in the system on the location page. A similar function already exists in the WSS for updating the mailing address in the [Update Personal Info](#) transaction.

## Creating an XAI Schema

This section assumes that you are familiar with the topics described in [XML Application Integration](#).

The main steps for our purpose are to go in the [XAI Schema Editor](#) to the “Schemas” menu item, and to create an XAI inbound service. Use the search function to assist you in finding the correct service. Our location maintenance is based on the Meta info file CILCPRMP.

The schema needs to be saved and registered. You should also [test the new service](#).

**Naming Convention.** In order to ensure that new files added in future releases do not override your files, be sure to use “CM” for “customer modification” for the first two letters of your new service. In addition, set the owner flag to **Customer Modification**.

## Creating a Java Bean

The java bean is responsible for retrieving information from the respective Oracle Enterprise Taxation Management object. Within the classes directory of the WSS application you will find two kinds of java classes: One group ending their class name with “Action” (these are discussed in the next section) and the other group ending with “Bean” (so called object beans).

You can take almost any java class ending with “Bean” as an example and template for developing a new bean.

**Note.** Not all classes with names ending with “Bean” are object classes with the characteristics mentioned below. Some of those beans serve only as container classes describing the data members of an Oracle Enterprise Taxation Management object, without having any additional logic to retrieve data via XAI. Usually such container classes are used when there are more than two data members in the object and will be referenced by an object class retrieving more than one record. For example, in the Account Info page we are displaying the obligations for an account. The object bean retrieving those obligations is the “SAforAccountBean” and it stores the obligations in a data array, whose members are instances of the container class “SAforAccount”.

The following is a description of the common parts of all object beans:

- All beans belong to the package com.splwg.selfservice.
- All beans communicating with XAI will need to import the package org.dom4j.\*.

- All beans implement the java class `java.io.Serializable`.
- Variables are always defined in private scope, and a public function exists to populate and/or to retrieve their values.
- The property object is used to hold all properties that exist in the `SelfServiceConfig.properties` file (e.g. the XAI server address). This object needs to be transferred from the action bean to the object bean. You may either achieve this by providing the property object as an argument to the constructor, or by explicitly setting it in a separate method.
- One main function contains the XAI request to be sent to the XAI server and the code to parse the response.

For this purpose it might be helpful to use the request and response xml, so that the elements and attributes can be copied into your code. You can see this xml by using the XAI Submission page or the XAI Dynamic Submission page for the appropriate XAI inbound service. In addition, if XAI logging is switched on, you may view the `xai.log` file to see all requests and responses received or sent by the XAI server. Another option is to use the XAI Schema Editor testing tool, in order to submit such a request to XAI.

All attributes within elements you may want to retrieve from the service need to get parsed and stored into variables.

Any error message returned (or created by the bean) should be stored into a variable (usually named "errorMsg"), and its value will be inquired by the calling method. Refer to [Error Message Handling](#) for more information.

For our test case example of creating a service for service address maintenance, you will likely want to create two main methods.

- One method to retrieve the data according to the primary id to be provided
- The other one to enable the update of the service address according to all attributes provided as input parameters.

**Naming Convention.** In order to ensure that new files added in future releases do not override your files, be sure to use "CM" for "customer modification" for the first two letters of every new file.

## Creating an Action Class

One kind of java class has an "Action" suffix and controls the process flow, which might be more or less complicated according to the required functionality.

This action class is always called from the controller class. Refer to [Control Classes](#) for more information.

The common characteristics of this type of class are:

- All action classes implement the interface `SSvcAction`.

This interface defines one common method by the name of "perform". The `HttpServletRequest`, `HttpServletRequest`, and `HttpServletRequestResponse` instances are required arguments, ensuring accessibility to the servlet properties and servlet session to every class.

- A transaction might have different steps. For example, the first step for the update personal info transaction is to show the current personal info data. The second step processes the updated data. According to the different steps, different methods are called. For an example of a multi-step transaction, refer to `PayOnlineAction.java`.
- An instance of the object bean is created and after setting the input parameters, the main function is called.
- According to the result of the previous step, a JSP page is called (by calling the forward method of the servlet request dispatcher).

For your new transaction, you must create an action class to execute the appropriate steps. You must also go to the Actions section of the `CMSelfServiceConfig.properties` file to indicate the name of the class along with its action (reference) name. This is the name used to reference the class in your object bean.

The `CMSelfServiceConfig.properties` file was configured at installation time. Refer to the installation documentation for more information. This file can be found in the `WEB-INF` directory of the `SelfService` directory in your server. Examples of how to define your action name may be found in this file. The property name used to define the property must start with `com.splwg.selfservice.Action`. The `SSvcController` class will dynamically load these actions in the `initActions` method.

## Control Classes

There are some classes that exist to perform a special task.

- The “`SSvcController`” class is the servlet called by all JSP forms on submission. The first time it is called is when starting the `SelfService` application. (The web application directory “`SelfService`” contains the file “`index.html`”, which refers the user to this servlet with a login action.)

According to the action called, the respective action class is invoked.

This class also keeps track of the servlet session instance, and if it has timed out, it forces the user to log on again.

- The “`XAIHTTPCall`” class administers the call to XAI. All object beans communicating with XAI call it. You won’t need to change anything in this class.
- The “`Util`” class contains static methods to be called from various classes in the application. Currently it contains only one method that replaces null values with blanks.
- The “`XMLEncoder`” class may be called to make sure that any input string complies with xml or http standards. For example, special characters like an ampersand are replaced.

Our service address maintenance action class will have two steps. The first step is called when selecting the transaction from the menu, and reads the respective location record by invoking the created object bean. The second step is called upon submission of the form, and processes the changes submitted.

## Creating a JSP Page

Every JSP page consists of three parts: the HTML part, the server script and the client script. All HTML elements are legitimate in a JSP page, and both the client script as well as the server script may dynamically control the display elements of the page.

This capability is used several times in the application. One example is the dynamic creation of lists, like the obligation list in the Account Information screen, or the address list in the Update Personal Info screen.

A server script using java may use all and classes available to it. Object beans are transferred from the action bean as attributes of either the session or the request object. Depending on this differentiation, the JSP routine `jsp:useBean` will have either the scope request or session.

Other JSP or html files may get included with the include command.

The client script contains mostly validation methods called before the form is submitted.

By sticking to the style sheet with its already defined styles and avoiding style definitions in HTML elements, you will keep graphically in line with the existing pages.

For the service address maintenance you will probably create two pages. One page displays all the data retrieved, where either all or some will be modifiable by the taxpayer. The other page informs the taxpayer that the data was updated successfully.

**Naming Convention.** In order to ensure that new files added in future releases do not override your files, be sure to use "CM" for "customer modification" for the first two letters of every new file.

## Modifying the Menu

A file called `Menu.jsp` contains a `DisplayMenu` function that determines what menu items to display. This function first checks whether or not a `CMDisplayMenu` function exists. If one does, that method is called to display the menu items. If no CM menu file exists, then the `CIDisplayMenu` function is used.

To customize the items listed in the menu, perform the following steps:

- Edit `CMMenu.jsp` (found in the `SelfService` directory) and modify the `CMDisplayMenu` method to add/remove menu contents as desired. For example, you may choose to hide a menu item that you do not want to provide for your taxpayers and/or you may add new menu items that refer to new pages you have created.

For each new JSP file that you have created, you must do the following:

- Include the files `Menu.jsp`, `CIMenu.jsp` and `CMMenu.jsp` into your JSP file.
- Call the `DisplayMenu` method from your JSP file. Refer to any of the JSP files provided by the system as an example.

## Style Sheet Modifications

JSP UI elements have the style defined in their class attribute. The declaration of all styles is found in the `wss.css` file within the `SelfService` directory. The JSP pages reference the style sheet by including a file called `IncPreamble.jsp`. When this JSP is loaded, it checks whether or not a `CMwss.css` file exists. If one does, that style sheet is used for the page. If no CM style sheet file exists, the page uses the original `wss.css`.

To customize the style sheet provided, perform the following steps:

- Make a copy of the `wss.css` file and call it `CMwss.css`.
- Make changes to `CMwss.css` as desired.

## WSS Standards

### Contents

- Error Message Handling
- Session Object Information

### Error Message Handling

The main methods in the object beans (the ones communicating with XAI) have boolean return values. The action bean calls those methods.

If the call to such a method returned true, the program flow continues. A return code of false informs the action bean that an error has occurred.

As a default, the error message that may be received from the Oracle Enterprise Taxation Management service or that may be created by the method is stored in a variable by the name of "errorMsg", and the action bean retrieves this variable.

Usually the action bean then redirects the program flow back to the calling JSP page.

Each JSP page includes a JSP file ("ErrorHandler.jsp") at the beginning of its form, which checks whether the transferred error message variable is unlike null, and if so displays a message box with the error message.

An error message returned from the server is displayed in the appropriate language defined for the self-service application. The Oracle Enterprise Taxation Management message should be displayed or trapped and mapped to a more user friendly message. Refer to NewPaymentBean.java for an example of overriding a message received by Oracle Enterprise Taxation Management.

All messages that are generated by a WSS object or an action bean are also stored in the system message table to take advantage of the multi-language storage of messages. The application displays the message in the appropriate language for the self-service application.

**New Error Messages.** If your new page requires new messages, define the messages in the system using message category 90000, which is reserved for implementations.

### Session Object Information

Data you want to transfer from an action bean to a JSP page may be put either into the http request object or into the http session object. The difference is in the scope.

If the information will only be valid for the specific request, then store the information in the request object. The session object stays active until the session is ended or timed out, and therefore any information you want to keep accessible in your application can be stored in the session object.

In our WSS application, the ValidUserBean is stored in the session object. On logon its information (account id, user name, entity name, currency symbol) is loaded and stays accessible by all WSS JSP pages.

# The Conversion Tool

This document describes the Oracle Enterprise Taxation Management conversion tool.

## Contents

- [Introduction](#)
- [Conversion Tool Steps](#)
- [The Validation User Interface](#)
- [The Staging Tables](#)
- [Appendix A - Entity Relationship Diagramming Standards](#)
- [Appendix B - Multiple Owners In A Single Database](#)
- [Appendix C - Known Oddities](#)

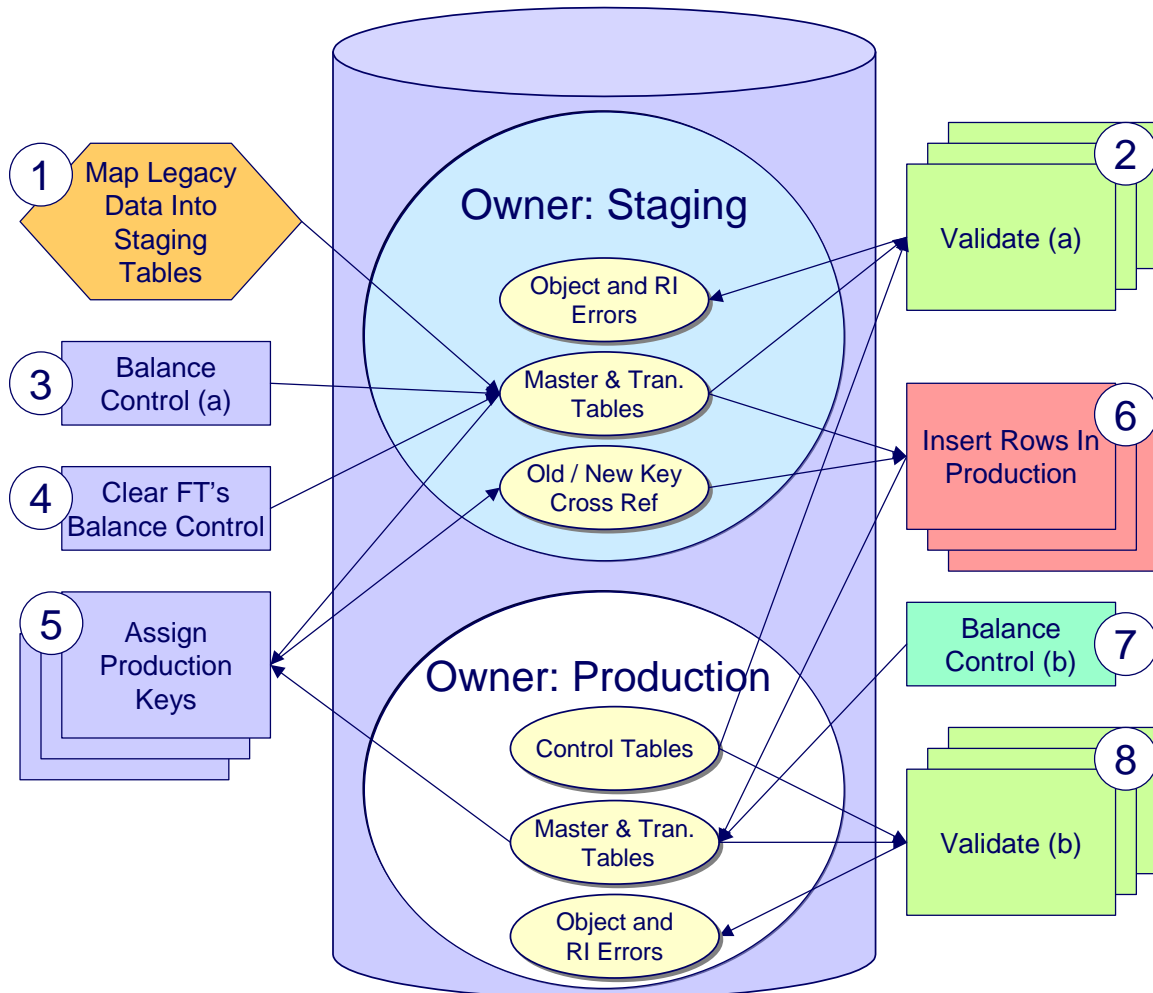
## Introduction

---

When you're ready to convert data from your legacy system into Oracle Enterprise Taxation Management, you will have analyzed your requirements according to your business and organizational needs and set up the control tables accordingly.

Refer to the **Administration Guide** for a complete discussion of the various control tables and the order in which they must be set up.

After the control tables are set up, you are ready to load data into the system from your legacy system. This conversion effort involves several steps as illustrated in the following diagram:



The following points briefly outline each of the above tasks:

- Map Legacy Data Into Staging.** During this step, your legacy master data (e.g., account, person, location) and transaction data (e.g., bills, payments) is migrated into the system. Notice that you are not migrating this data directly into production. Rather, your rows are loaded into tables that are identical to the production tables; they just have a different owner. Refer to [Appendix B – Multiple Owners In A Single Database](#) for information about table ownership.

**Different Databases For Staging and Production.** The above diagram illustrates how the system is configured to support the conversion effort in the standard installation, i.e., the staging tables are in the same database as the production tables (each with a different owner). However, it is possible for the staging tables to be in a separate database. This option requires additional effort on your part (because you would have to copy the control tables from production into your staging database). Please refer to [Appendix B – Multiple Owners In A Single Database](#) for information about this alternative.

Mapping legacy data into the system is probably the most challenging part of the conversion process because the system is a normalized database (and most legacy applications are not).

- **Validate (a).** During the validation (a) step, the system validates the data you loaded into the staging tables. Two types of validation programs exist:
  - **Object Validation Programs.** Each of the system's master data objects (e.g., person, account, location, etc.) is validated using the same logic that is used to validate data added by users in your production system.
  - **Referential Integrity Validation Programs.** After you have successfully validated the master data objects, the referential integrity validation programs are executed to validate transaction data and to highlight "orphaned" rows. These programs check the validity of the foreign keys on all rows on all tables.

**Control tables from production.** It's important to notice that the validation programs validate your staging data using the control tables that have been set up in production. Refer to [Appendix B – Multiple Owners In A Single Database](#) for a description of how this works.

- **Balance Control (a).** During this step, you run the balance control program and then verify that the balances that it generates are consistent with the balances in your legacy system.
- **Clear FT's Balance Control.** In the previous step, the system creates a balance control and links it to the FT's. If the balance control's balances are consistent with the amount of receivables being transferred into the system, you should run the Clear FT's Balance Control program. This program simply resets the Balance Control column on the FT so that the FT's can be included in a balance control (see the last step below) after they have been transferred to production.
- **Assign Production Keys.** During this step, the system allocates random, clustered keys to the rows in the staging database.
- **Insert Rows Into Production.** During this step, the system populates your production tables with rows from the staging. When the rows are inserted, their prime keys are reassigned using the data populated in the previous step.
- **Balance Control (b).** During this step, you run the balance control program against production. You do this to verify the balances in production are consistent with the values of receivables converted from your legacy application.
- **Validate (b).** During this step, you rerun the object validation programs, but this time against production. We recommend rerunning these programs to confirm that the insertion programs have executed successfully. We recommend running these programs in random sample mode (e.g., validate every 1000<sup>th</sup> object) rather than conducting a full validation in order to save time. However, if you have time, you should run these programs in full validation mode (to validate every object).

## Conversion Tool Steps

---

The following sections provide more details about the steps in the conversion process.

### Contents

- [Map Legacy Data Into Staging Tables](#)
- [Validate Information In The Staging Tables](#)
- [Balance Control \(a\)](#)



Clear FT Balance Control  
Allocate Production Keys  
Insert Production Data  
Run Balance Control Against Production  
Validate Production

## Map Legacy Data Into Staging Tables

This section provides some high level discussion about mapping legacy data to the system's staging tables. Refer to [The Staging Tables](#) for details about the staging tables in the system.

**Recommendation.** You can use any method you prefer to load Oracle Enterprise Taxation Management data from your legacy application. However, we recommend that you investigate your database's mass load utility (as opposed to using insert statements) as the mechanism to load the staging tables. In addition, we strongly recommend that you disable the indexes on these tables before populating these tables and then enable the indexes after populating these tables.

### A Note About Keys

The prime keys of the tables in the staging database are either system-assigned random numbers or they aren't. Those tables that don't have system-assigned random numbers have keys that are a concatenation of the parent's prime-key plus one or more additional fields.

Every table whose prime key is a system-assigned random number has a related table that manages its keys; we refer to these secondary tables as "key tables". The following points provide more information about the key tables:

- Key tables are used by programs that allocate new keys. For example, before a new account ID is allocated, the key assignment program checks the account key table to see if it exists.
- Key tables exist to support archiving and ConfigLab requirements.
  - When an object is [archived](#), its row is removed from the primary table, but its key remains on the key table. This is done so that the key isn't reused in production, as we want to be able to reinstate an archived object.
  - From a [ConfigLab](#) perspective, these key tables are used to prevent a key from being reused in production for an object added in a ConfigLab. For example, a user might add an account in a ConfigLab environment and we don't want its key to be allocated to an account added in production.
- Key tables only have two columns:
  - The key of the object.
  - An environment ID. The environment ID identifies the database in which the object resides.
- Key tables are named the same as their primary table with a suffix of "\_K". For example:
  - The key table for CI\_ACCT is CI\_ACCT\_K
  - The key table for CI\_PREM is CI\_PREM\_K

The name of every table's key table is defined under the Generated Keys column in the Table Names sections in [The Staging Tables](#).

- When you populate rows in tables with system-assigned keys, you must also populate a row in the related key table. For example, if you insert a row into CI\_ACCT, you must also insert a row into CI\_ACCT\_K. The environment ID of these rows must be the same as the environment ID on this database's [installation record](#).
- When you insert rows into your staging database, the keys do not have to be random, system-assigned numbers. They just have to be unique. A later process, [Allocate Production Keys](#), will allocate random, system-assigned keys prior to production being populated.

## Validate Information In The Staging Tables

During the first validation step, the system validates the data you loaded into the staging tables. Two types of validation programs exist:

- **Object Validation Programs.** The object validation programs validate each of the system's master data objects (e.g., person, account, location, etc.) and a limited number of transaction data objects (e.g., billable charge). Please note that these programs call the same programs that are used to validate data added by users in your production system.
- **Referential Integrity Validation Programs.** After the master data objects have been validated, the referential integrity validation programs are executed to validate transaction data and to highlight "orphaned" rows. These programs simply check the validity of the foreign keys on all rows on all tables.

The contents of this section describe how to execute the validation programs.

### Contents

[Object Validation Programs](#)  
[Submitting Object Validation Programs](#)  
[Referential Integrity Validation Programs](#)  
[Submitting Referential Integrity Validation Programs](#)  
[Recommendations To Speed Up Validation Programs](#)

## Object Validation Programs

Each of the objects described under [Master Data](#) must be validated using the respective object validation program indicated in its Table Names section.

In a limited number of cases object validation is available for [Transaction Data](#) objects, where customers may convert transaction data that is still pending. For the same objects you may also be converting historic records. You may not want to perform validation on completed records. As a result the background processes provided for transaction data allow you to limit the validation to records in a given status.

We strongly recommend validating each object in the following steps:

- Execute each object's validation program in random-sample mode to highlight pervasive errors. When you execute a validation in random-sample mode, you are actually telling it to validate every X records (where X is a parameter that you supply to the job). Refer to [Submitting Object Validation Programs](#) for more information about the parameters supplied to these background processes.

- You can view errors highlighted by validation programs using the [Validation Error Summary](#) transaction.
- Correct the errors using SQL. Note, you can use the base package's transactions to correct an error if the error isn't so egregious that it prevents the object from being displayed on the browser.
- After all pervasive errors have been corrected; re-execute each object's validation program in all-instances mode to highlight elusive, one-off errors. Refer to [Submitting Object Validation Programs](#) for more information about the parameters supplied to these background processes.

**Take note!** Whenever an object validation program is run, it is necessary to delete all previously recorded errors associated with its tables from the validation error table before it inserts new errors.

After the various object validation programs run cleanly, run the referential integrity validation programs as described in the next section.

## Submitting Object Validation Programs

The object validation programs that are described in the [staging tables](#) table names matrices are classic background processes as they can also be run against production data. You submit these processes in the same way you submit any background process in production. Refer to [Object Validation Processes](#) for information about these processes and their parameters.

## Referential Integrity Validation Programs

It's important to understand that only master data objects (e.g., persons, accounts, assets, locations, etc.) are validated by the object validation programs discussed above. This means that only master data objects will have their foreign keys checked for validity by the object validation programs. You must run the referential integrity programs to validate all other data.

The referential integrity validation programs highlight:

- Orphaned rows because orphan rows, by definition, don't reference an object.
- Invalid foreign keys on transaction data.

**Validating Transaction Data.** You may wonder why transaction data is not subject to the object validation routines. This is because: a) the production system only needs validation logic for master data because transaction data is not entered by users, and b) most conversions necessitate loading skeletal transaction data because the legacy system typically doesn't contain enough information to create accurate transactions in the system.

Each of the tables described under [Transaction Data](#) must be validated using the respective referential integrity validation program indicated in its Table Names section. We strongly recommend validating each table in the following steps:

- Execute each table's referential integrity validation program. Refer to [Submitting Referential Integrity Validation Programs](#) for more information about the parameters supplied to these background processes.

- You can view errors highlighted by this process using the [FK Validation Summary](#) transaction.
- Correct the errors using SQL (you cannot use the application to correct these types of errors).
- Rerun the referential integrity programs until no errors are produced.

**Take note!** Whenever you run a referential integrity validation program, it deletes all errors associated with its table from the referential integrity error table.

In order to highlight orphaned rows in the master data, run the referential integrity validation programs against all tables described under [Master Data](#) using the procedure described above.

## Submitting Referential Integrity Validation Programs

The referential integrity validation programs described under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices) are submitted using a batch driver program, **CIPVRNVB**, and this program is executed in the staging database. Please note that the referential integrity validation programs may also be run in the production environment on occasion, to determine the integrity of data in the production database.

Refer to [Referential Integrity Validation Processes](#) for information about these processes and their parameters.

You should supply the following parameters to this program:

- **Batch code.** The batch code associated with the appropriate table's referential integrity validation program. Refer to each table listed under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices) for each referential integrity batch code / program.
- **Batch thread number.** Thread number is not used and should be left blank.
- **Batch thread count.** Thread count is not used and should be left blank.
- **Batch rerun number.** Rerun number is not used and should be left blank.
- **Batch business date.** Business date is the date supplied to the referential integrity validation programs and the date under which statistics will be logged.
- **Total number of commits.** Total number of commits is not used and should be left blank.
- **Maximum minutes between cursor re-initiation.** Maximum minutes between cursor re-initiation is not used and should be left blank.
- **User ID.** User ID is only used to log statistics for the execution of the batch job.
- **Password.** Password is not used.
- **Language Code.** Language code is used to access language-specific control table values. For example, error messages are presented in this language code.
- **Trace program at start (Y/N), trace program exit (Y/N), trace SQL (Y/N) and output trace (Y/N).** These switches are only used during QA and benchmarking. If trace program start is set to Y, a message is displayed whenever a program is started. If trace program at exist is set to Y, a message is displayed whenever a program is exited. If trace SQL is set to Y, a message is displayed whenever an SQL statement is executed.

## Recommendations To Speed Up Validation Programs

The following points describe ways to accelerate the execution of the validation programs:

- Ensure that statistics are recalculated after data has been inserted into the staging tables. For Oracle users, we strongly recommend using the Oracle-provided PL/SQL package to generate statistics rather than the analyze command.
- [Object validation programs](#) should be run multi threaded.
- Execute shorter running validation processes (e.g., less records) first so that the error data can be analyzed while other processes are busy running.
- [Referential integrity validation programs](#) run fairly quickly without much tuning. However, additional benefits are gained by running several programs at the same time.
- Remember that the [object validation programs](#) can be run in “validate every n<sup>th</sup> row”. We recommend running these programs using a largish value for this parameter until the pervasive problems have been rectified.

## Balance Control (a)

During this step, you run the balance control programs and then verify that the balances that it generates are consistent with the balances in your legacy system.

**Submitting this process.** You submit this process in the staging database. Refer to [The Big Picture of Balance Control](#) for more information about the balance control processes. Refer to [Balance Control](#) for information about the page used to view the balances generated by this process.

## Clear FT Balance Control

In the previous step, the system created a balance control and links it to the FT's. If the balance control's balances are consistent with the amount of receivables being transferred into the system, you should run the Clear FT's Balance Control program. This program simply resets the Balance Control column on the FT so that the FT's can be included in a balance control (see the last step below) after they have been transferred to production. Note: the batch control ID of **CNV-BCG** is used to request this process.

**Submitting this process.** You submit this process in the staging database. Refer to [Reset Balances](#) for a description of this process and its parameters.

## Allocate Production Keys

The topics in this section describe the background processes used to assign production keys to the staging data.

### Contents

- [The Big Picture of Key Assignment](#)
- [Submitting Key Assignment Programs](#)

## Recommendations To Speed Up Key Generation Programs

### The Big Picture of Key Assignment

It's important to understand that the system does not overwrite the prime-keys on the rows in the staging database, as this is a very expensive IO transaction. Rather, a series of tables exist that hold each row's old key and the new key that will be assigned to it when the row is [transferred into the production database](#). We refer to these tables as the "old key / new key" tables. The old key / new key tables are named the same as their primary table, but rather than being prefixed by "CI", they are prefixed by "CK". For example, the old key / new key table for CI\_ACCT is called CK\_ACCT.

The key assignment programs listed under [Master Data](#) and [Transaction Data](#) (in the table names sections) are responsible for populating the old key / new key tables (i.e., you don't have to populate these tables). Because the population of the rows in these tables is IO intensive, we have supplied detailed instructions that will accelerate the execution time of these programs.

**Why are keys reassigned?** The conversion process allocates new prime keys to take advantage of the system's parallel processing and data-clustering techniques in the production system (these techniques are dependent on randomly assigned, clustered keys).

**Iterative conversions.** Rather than perform a "big bang" conversion (one where all customers are populated at once), some implementations have the opportunity to go live on subsets of their customer base. If this describes your implementation, please be aware that the system takes into account the existing prime keys in the production database before it allocates a new key value. This means when you convert the next subset of customers, you can be assured of getting clean keys.

**Program Dependencies.** The programs used to assign production keys are listed in the Table Names matrices. Most of these programs have no dependencies (i.e., they can be executed in any order you please). The exceptions to this statement are noted in [Program Dependencies](#).

### Submitting Key Assignment Programs

The key assignment programs described under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices) are submitted using a batch driver program, **CIPVRNKB**, and this program is executed in the staging database. You should supply the following parameters to this program:

- **Batch code.** The batch code associated with the appropriate table's key assignment program. Refer to each table listed under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices) for each key assignment batch code / program.
- **Batch thread number.** Thread number is not used and should be left blank.
- **Batch thread count.** Thread count is not used and should be left blank.
- **Batch rerun number.** Rerun number is not used and should be left blank.
- **Batch business date.** Business date is the date supplied to the key assignment programs and the date under which statistics will be logged.

- **Total number of commits.** Total number of commits is not used and should be left blank.
- **Maximum minutes between cursor re-initiation.** Maximum minutes between cursor re-initiation is not used and should be left blank.
- **User ID.** User ID is only used to log statistics for the execution of the batch job.
- **Password.** Password is not used.
- **Language Code.** Language code is used to access language-specific control table values. For example, error messages are presented in this language code.
- **Trace program at start (Y/N), trace program exit (Y/N), trace SQL (Y/N) and output trace (Y/N).** These switches are only used during QA and benchmarking. If trace program start is set to Y, a message is displayed whenever a program is started. If trace program at exist is set to Y, a message is displayed whenever a program is exited. If trace SQL is set to Y, a message is displayed whenever an SQL statement is executed.
- **Mode.** The proper use of this parameter will greatly speed up the key assignment step as described under [Recommendations To Speed Up Key Generation](#). This parameter has three values:
  - If you supply a mode with a value of **I** (initial key generation), the system allocates new keys to the rows in the staging tables (i.e., it populate the respective old key / new key table).
  - If you supply a mode with a value of **D** (resolve duplicate keys), the system reassigns keys that are duplicates.
  - If you supply a mode with a value of **B** (both generate keys and resolve duplicates), the system performs both of the above steps. This is the default value if this parameter is not supplied.
  - Please see [Recommendations To Speed Up Key Generation](#) for how to use this parameter to speed up the execution of these processes.

**Parallel Key Generation.** Note well, no key generation program should be run (either in mode **I** or **B**) while another program is being run unless that program is in the same tier (see [Program Dependencies](#) for a description of the tiers).

- **Start Row Number.** This parameter is only used if you are performing conversions where data already exists in the tables in the production database (subsequent conversions). In an Oracle database the key assignment routines create the initial values of keys by manipulation of the Oracle row number, starting from 1. After any conversion run, a subsequent conversion run will start with that row number again at 1, and the possibility of duplicate keys being assigned will be higher. The purpose of this parameter is to increase the value of row number by the given value, and minimize the chance of duplicate key assignment.

## Recommendations To Speed Up Key Generation Programs

The following points describe ways to accelerate the execution of the key generation programs.



**Naming convention.** The convention “CK\_<table\_name>” is used to denote the old key / new key tables described under [The Big Picture of Key Assignment](#).

- Make the size of your rollback segments large. The exact size is dependent on the number of rows involved in your conversion. Our research has shown that processing 7 million rows generates roughly 3GB of rollback information.
  - Setup the rollback segment(s) to about 10GB with auto extend to a maximum size of 20GB to determine the high water mark
  - A next extent value on the order of 100M should be used.
- Make sure to turn off all small rollback segments (otherwise Oracle will use them rather than the large rollback segments described above).
- After the key assignment programs execute, you can reclaim the space by:
  - Keep a low value for the “minimum extent” parameter for the rollback.
  - Shrink the rollback segments and the underlying data files at the end of the large batch jobs.
- Compute statistics on the CK\_<table\_name> tables after every 50% increase in table size. Key generation is performed in tiers or steps because of the inheritance dependency between some tables and their keys. Although key generation for the tier currently being processed is performed by means of set-based SQL, computation of statistics between tiers will allow the database to compute the optimum access path to the keys being inherited from the **previous** tier’s generation run.  
For Oracle users, we strongly recommend using the Oracle-provided PL/SQL package to generate statistics rather than the analyze command.
- Optimal use of the **Mode** parameter under [Submitting Key Assignment Programs](#).
  - Before any key assignments, alter both the “old key” CX\_ID index and the “new key” CI\_ID index on the CK\_<table\_name> tables to be unusable.
  - Run all [key assignment tiers](#), submitting each job with MODE = “I”.
  - Rebuild the CX\_ID and CI\_ID indexes on the CK\_<table\_name>. Rebuilding the indexes using both the PARALLEL and NOLOGGING parameters will speed the index creation process in an Oracle DB. Statistics should be computed for these indexes.
  - Run all key assignment tiers that were previously run in MODE = ‘I’, submitting each job with MODE = “D”. This will reassign all duplicate keys.

## Insert Production Data

The topics in this section describe the background processes used to populate the production database with the information in the staging database.

### Contents

[The Big Picture Of Insertion Programs](#)  
[Submitting Insertion Programs](#)  
[Recommendations To Speed Up Insertion Programs](#)



## The Big Picture Of Insertion Programs

All insertion programs are independent and may run concurrently. Also note, all insertion programs can be run in many parallel threads as described in the next section (in order to speed execution).

## Submitting Insertion Programs

The insertion programs described under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices) are submitted using a batch driver program, **CIPVRNIB**, and this program is executed in the staging database. You should supply the following parameters to this program:

- **Batch code.** The batch code associated with the appropriate table's insertion program. Refer to each table listed under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices) for each insertion batch code / program.
- **Batch thread number.** Thread number contains the relative thread number of the process. For example, if you want to insert accounts into production in 20 parallel threads, each of the 20 execution instances receives its relative thread number (1 through 20). Refer to [Parallel Background Processes](#) for more information.
- **Batch thread count.** Thread count contains the total number of parallel threads that have been scheduled. For example, if the account insertion process has been set up to run in 20 parallel threads, each of the 20 execution instances receives a thread count of 20. Refer to [Parallel Background Processes](#) for more information.
- **Batch rerun number.** Rerun number is not used and should be left blank.
- **Batch business date.** Business date is the date supplied to the insertion programs and the date under which statistics will be logged.
- **Total number of commits.** This is the number of commits IN TOTAL that you want to perform. For example, if you have 1,000,000 accounts and you supply a value of **100**, then a commit will be executed for approximately every 10,000 accounts.
- **Maximum minutes between cursor re-initiation.** This should only be populated if you want to override the default value of **15**.
- **User ID.** User ID is only used to log statistics for the execution of the batch job.
- **Password.** Password is not used.
- **Language Code.** Language code is used to access language-specific control table values. For example, error messages are presented in this language code.
- **Trace program at start (Y/N), trace program exit (Y/N), trace SQL (Y/N) and output trace (Y/N).** These switches are only used during QA and benchmarking. If trace program start is set to Y, a message is displayed whenever a program is started. If trace program at exist is set to Y, a message is displayed whenever a program is exited. If trace SQL is set to Y, a message is displayed whenever an SQL statement is executed.

## Recommendations To Speed Up Insertion Programs

The following points describe ways to accelerate the execution of the insertion programs:

- Before running the first insertion program:

- Rebuild the index on the prime key on the old key / new key table (i.e., those tables prefixed with "CK").
- Re-analyze the statistics on the old key / new key table (i.e., those tables prefixed with "CK"). For Oracle users, we strongly recommend using the Oracle-provided PL/SQL package to generate statistics rather than the analyze command.
- Alter all indexes on the production tables being inserted into to be unusable.
- After the insertion programs have populated production data, rebuild the indexes and compute statistics for these tables. For Oracle users, we strongly recommend using the Oracle-provided PL/SQL package to generate statistics rather than the analyze command.

## Run Balance Control Against Production

During this step, you rerun the balance control program, but this time against production. You do this to verify the balances in production are consistent with the values of receivables converted from your legacy application.

**Submitting this process.** You submit this process in the production database. Refer to [The Big Picture of Balance Control](#) for more information about the balance control processes. Refer to [Balance Control](#) for information about the page used to view the balances generated by this process.

## Validate Production

During this step, you rerun the [object validation programs](#), but this time in production. We recommend rerunning these programs to confirm that the insertion programs have executed successfully. We recommend running these programs in random sample mode (e.g., validate every 1000<sup>th</sup> object) rather than conducting a full validation in order to save time. However, if you have time, you should run these programs in full validation mode (to validate every object). Please refer to the various "Table Names" sections above for the respective names of the programs to run.

## The Validation User Interface

---

The topics in this section describe the various pages that assist in the conversion effort.

### Contents

- [Validation Error Summary](#)
- [Validation Error Detail](#)
- [FK Validation Summary](#)
- [FK Validation Detail](#)

## Validation Error Summary

Navigate to the **Admin** menu and open **Validation Error Summary** to view validation errors associated with the objects defined in [Master Data](#).

### Description of Page

You can use **Table Name** to restrict errors to a specific object. If this field is left blank, all errors on all objects will be displayed.

The grid contains a separate row for each type of error. The following information is displayed:

- **Table Name** is the name of the main table associated with the object.
- **Message Category** and **Message Number** define the type of error. These fields are the unique identifier of the message that describes the error (the verbiage of this message is displayed in the **Message Text** column).
- **Count** contains the number of records with this error. Press the Go To button to see the individual records with the error.

## Validation Error Detail

This page is used to view validation errors of a given type associated with one of the objects defined in [Master Data](#). This transaction is not intended to be invoked from the **Admin** menu. Rather, drill into the validation details from [Validation Error Summary](#).

### Description of Page

Use **Table Name**, **Message Category**, and **Message Number** to define the object and the type of error you wish to display. The grid contains a separate row for each object with the given type of error. The following information is displayed:

- **Table Name** is the name of the main table associated with the object.
- **Record Identifier** is the unique identifier of the object with the error (e.g., the person ID, the account ID, the location ID, etc.). Press the Go To button to transfer to the maintenance page associated with the object.
- **Message Category** and **Message Number** define the type of error. These fields are the unique identifier of the message that describes the error (the verbiage of this message is displayed in the **Message Text** column).

## FK Validation Summary

Navigate to the **Admin** menu and open **FK Validation Summary** to view foreign key validation errors associated with the objects defined in [Master Data](#).

### Description of Page

You can use **Table Name** to restrict errors to a specific object. If this field is left blank, all errors on all objects will be displayed.

The grid contains a separate row for each type of error. The following information is displayed:

- **Table Name** is the name of the main table associated with the object.
- **Count** contains the number of records on this table that have this error.
- **Foreign Key Field Names 1 to 6** contain the names of the foreign keys contained on this table that have been found to be in error.

- **Foreign Key Values 1 to 6** contain the values within the foreign key fields that are found to be in error.

## FK Validation Detail

This page is used to view foreign key validation errors of a given type associated with one of the objects defined in [Master Data](#). This transaction is not intended to be invoked from the **Admin** menu. Rather, drill into the validation details from [FK Validation Summary](#).

### Description of Page

Use **Table Name** to specify the table you wish to view. The names and values of the foreign key fields on the table are displayed. The grid that follows contains the primary key values of this table's records that are in error. The following information is displayed:

- **Table Name** is the name of the main table.
- **FK Fields 1 to 6** are the names of the foreign keys contained in this table. Displayed alongside the key names are the values within these fields. These identify records on other tables to which this table's record is related. For example, the CI\_PREM\_GEO record identified by its displayed primary keys should be related to a Location record with the Location ID shown – it appears in this list only if there is something amiss with this relationship.

## The Staging Tables

This section describes the objects into which your legacy data is mapped. For each object, we provide the following:

- A data model.
- An indication of which tables have system-assigned keys.
- The physical table names.
- The name of the batch control to submit to validate the object.
- The name of the program (and related batch control) that validates each table for referential integrity.
- The name of the program (and related batch control) that performs key assignment for each table.
- The name of the program (and related batch control) that inserts the table's rows into production from staging.
- Suggestions to assist in the conversion process.

**Recommendation.** We recommend you read this document on a browser (or using Word under windows) so you can take advantage of the [Color Coding](#).

**Column details do not appear in this document.** When you're ready to examine an object's tables, use the hyperlinks in the respective Table Names section to transfer to the [data dictionary](#). The data dictionary will show you the required columns, the foreign keys (and their related tables), the source code of the program that validates the contents of the table, and a host of other information that will assist the conversion process.

**Look Up and Control Tables.** In the data models that appear below, you will find a variety of entities that are classified as either a control table or a lookup table. Please refer to [Color Coding](#) for more information about how to recognize such an entity.

## Contents

[A Note About Programs in the Table Names Matrices](#)  
[Master Data](#)  
[Transaction Data](#)  
[Program Dependencies](#)

## A Note About Programs in the Table Names Matrices

For each object described in the master data and transaction data sections, there is a "table names" section that includes a matrix listing the name of each table that is part of the maintenance object. Included in the matrix is information about the programs provided to perform object validation, referential integrity validation, key assignment and insertion. The following are some points about these programs:

- One object validation program exists for the entire set of tables for the maintenance object. The **Object Validation Batch Control** column indicates the batch control used to submit the object validation. Refer to [Submitting Object Validation Programs](#) for more information. Drilling down on the hypertext allows you to see more information about the batch control, including the program associated with it.
- A referential integrity validation program exists for every table whose key includes a parent key of another object. As described in [Submitting Referential Integrity Validation Programs](#), these programs are submitted using a driver supplied by the system where the batch code for the appropriate table is provided. (The driver then executes the program whose name matches the batch code). The **Referential Integrity Validation Batch Control** column indicates the table's batch control / program name.
- One key assignment program exists for the parent table for the maintenance object. As described in [Submitting Key Assignment Programs](#), these programs are submitted using a driver supplied by the system where the batch code for the appropriate table is provided. (The driver then executes the program whose name matches the batch code). The **Key Assignment Batch Control** column indicates the table's batch control / program name.
- An insertion program exists for every table for the maintenance object. As described in [Submitting Insertion Programs](#), these programs are submitted using a driver supplied by the system where the batch code for the appropriate table is provided. (The driver then executes the program whose name matches the batch code). The **Insertion Batch Control** column indicates the table's batch control / program name.

## Master Data

This section describes the various “master data” objects (e.g., person, account etc.) that must be created before you can convert transaction data.

**Key Assignment Dictates The Order Of Conversion.** The following contents are listed in the order in which the objects should be converted in order to maintain referential integrity.

### Contents

- [Person](#)
- [Account](#)
- [Location](#)
- [Obligation](#)
- [Tax Role](#)

## Person

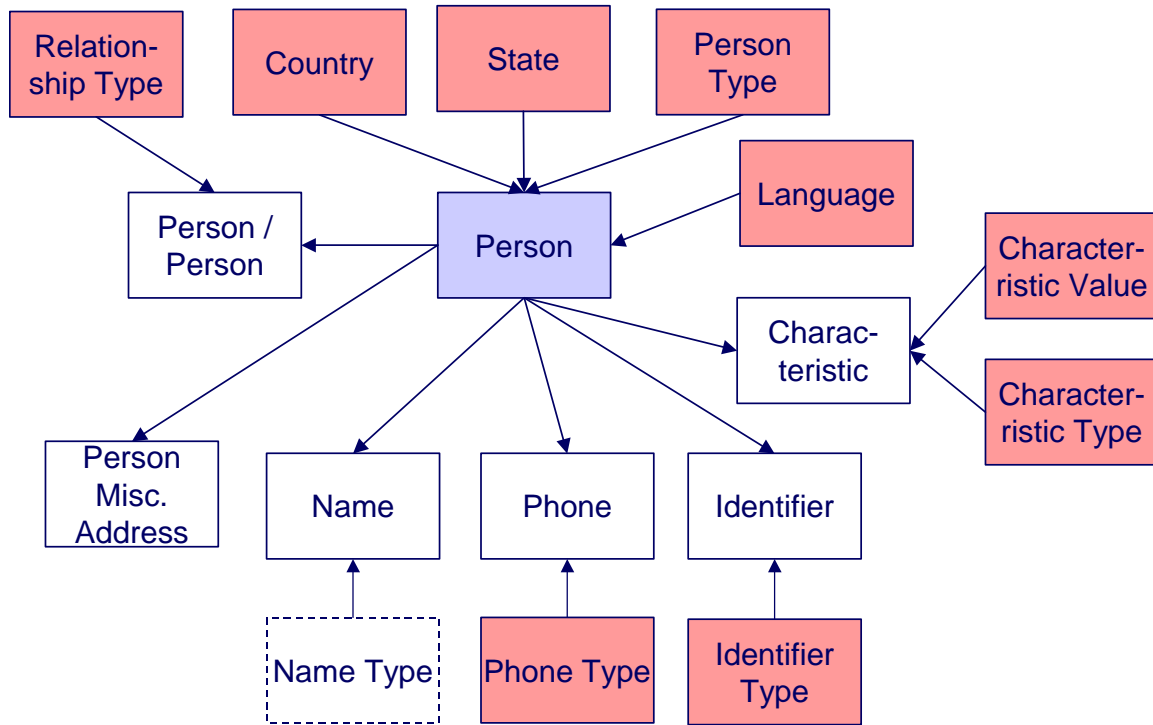
This section describes the person object. Refer to [Account](#) for details about the account object.

### Contents

- [Person Data Model](#)
- [Person Table Names](#)
- [Person Suggestions](#)

### Person Data Model

The following data model illustrates the person object.



### Person Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Person	<a href="#">CI_PER</a>	Yes <a href="#">CI_PER_K</a>	<a href="#">VAL-PER</a>		CIPVPERK	CIPVPERI
Name	<a href="#">CI_PER_NAME</a>	No. The key is PER_ID plus a sequence number.		CIPVPMNV		CIPVPMNI
Person / Person	<a href="#">CI_PER_PER</a>	No. The key is PER_ID1, PER_ID2, relationship type and start date.		CIPVPPEV		CIPVPPEI
Phone	<a href="#">CI_PER_PHONE</a>	No. The key is PER_ID plus phone type.		CIPVPPHV		CIPVPPHI
Identifier	<a href="#">CI_PER_ID</a>	No. The key is PER_ID plus identifier type.		CIPVPIDV		CIPVPIDI

Characteristic	<a href="#">CI_PER_CHAR</a>	No. The key is PER_ID plus an edate and a char type.		CIPVPRCV		CIPVPRCI
Miscellaneous Address	<a href="#">CI_PER_ADDR_SEAS</a>	No. The key is PER_ID plus a sequence number.		CIPVPSAV		CIPVPSAI

### Person Suggestions

A person must have at least one row on the name table and at least one of the names must be marked as being the primary name.

A person must have at least one row on the identity table and at least one of the identities must be marked as being the primary ID.

The country and state are only necessary if the person has an override mailing address.

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

## Account

Each customer must have a person and an account object. This section describes the account object, refer to [Person](#) for details about the person object.

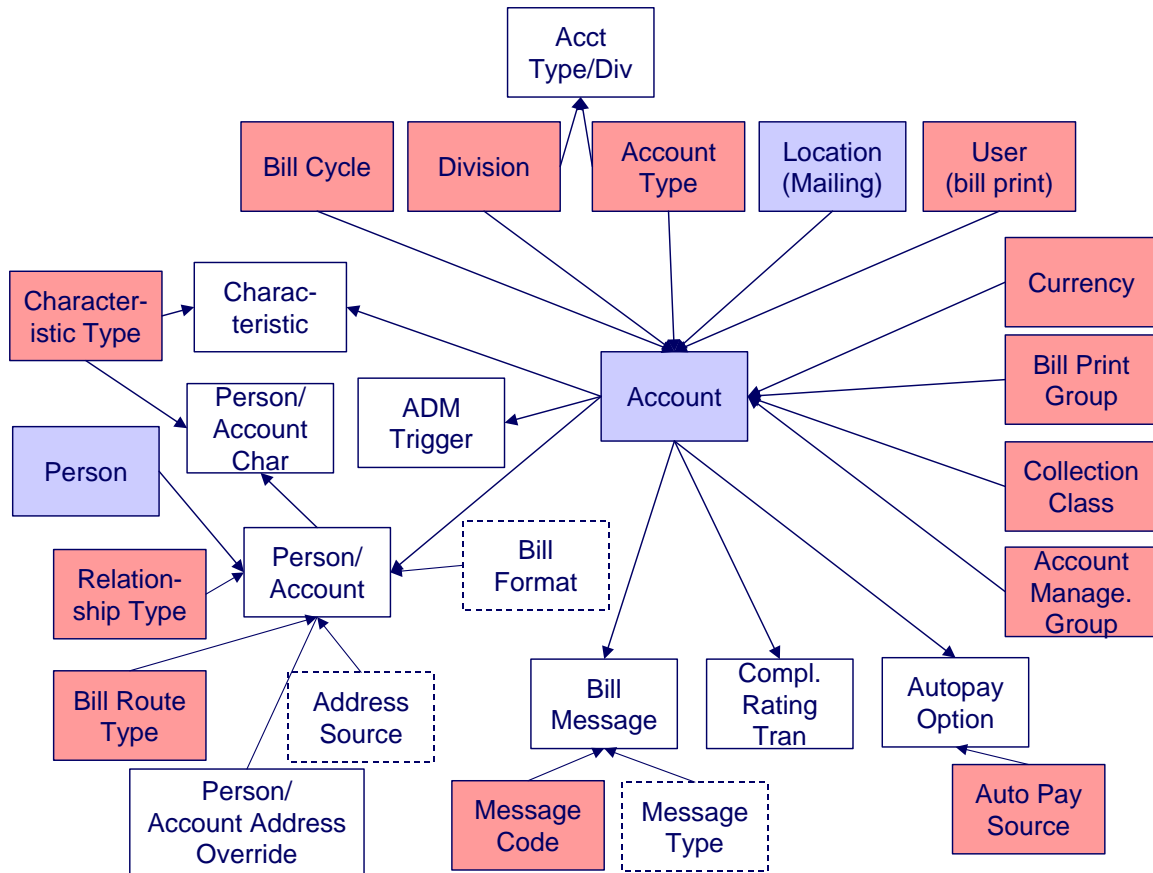
### Contents

- [Account Data Model](#)
- [Account Table Names](#)
- [Account Suggestions](#)

### Account Data Model

The following data model illustrates the account object.





## Account Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Account	<a href="#">CI_ACCT</a>	Yes <a href="#">CI_ACCT_K</a>	<a href="#">VAL-ACCT</a>		CIPVACCK	CIPVACCI
Bill Message	<a href="#">CI_ACCT_MSG</a>	No. The key is account ID plus bill message code.		CIPVMSGV		CIPVMSGI
Autopay Option	<a href="#">CI_ACCT_APAY</a>	Yes <a href="#">CI_ACCT_APAY_K</a>		CIPVAAPV	CIPVAAPK <a href="#">Has dependencies</a>	CIPVAAPI
Characteristic	<a href="#">CI_ACCT_CHAR</a>	No. The key is ACCT_ID plus an edate and a char type.		CIPVACHV		CIPVACHI
Person/Account	<a href="#">CI_ACCT_PER</a>	No. The key is account ID plus person ID.		CIPVACPV		CIPVACPI

Person / Account Char	<a href="#">CI_ACCT_PER_CHAR</a>	No. The key is account ID plus person ID plus an edate and a char type.		CIPVAPCV		CIPVAPCI
Person/Account Address Override	<a href="#">CI_PER_ADDR_OVERRIDE</a>	No. The key is Account ID plus Person ID		CIPVPAOV		CIPVPAOI
Compliance Rating Transaction	<a href="#">CI_CR_RAT_HIST</a>	Yes <a href="#">CI_CR_RAT_HIST_K</a>		CIPVCRTV	CIPVCRRK <a href="#">Has dependencies</a>	CIPVCRTI
ADM Trigger	<a href="#">CI_ADM_RVW_SCH</a>	No. The key is account ID plus date		CIPVARSV		CIPVARSI

### Account Suggestions

An account must have at least one row on the account / person table and at least one account / person must be marked as being the main customer. Please see column notes for the account / person table for inter-field validation in respect of the various switches (e.g., if main customer switch is on, then the person must also be financially responsible).

We recommend storing an ADM trigger ([CI\\_ADM\\_RVW\\_SCH](#)) for every account where the trigger date is the conversion date. This will cause the account to be reviewed by the [overdue monitor](#) when it next runs. We have supplied a dedicated batch process for this purpose that simply inserts a row in this table with the review date set equal to the current date. This will ensure that all converted accounts are reviewed after they are inserted into production. This program is named CIPVADMB and goes by the batch control ID of **CNV-ADM**.

If your legacy system has the equivalent of a compliance rating, you should create compliance rating transactions. The values you create need to be consistent with the base and threshold compliance rating on the installation record. Refer to the account user documentation for more information.

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

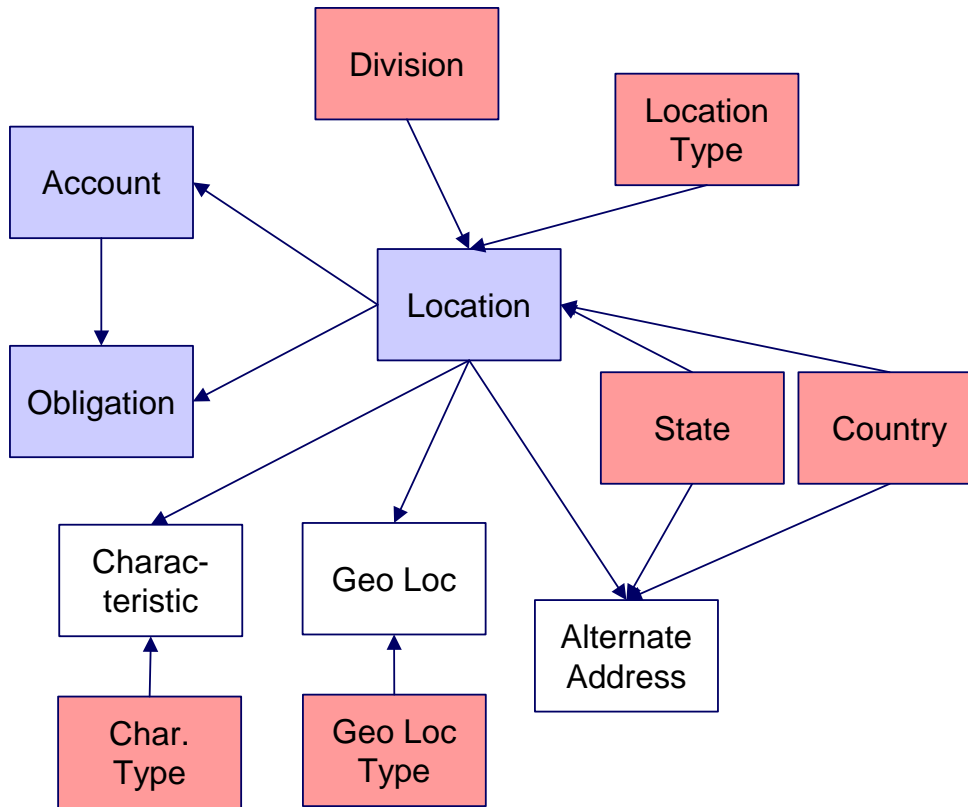
## Location

### Contents

- [Location Data Model](#)
- [Location Table Names](#)
- [Location Suggestions](#)

### Location Data Model

The following data model illustrates the location object.



Location Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Location	<a href="#">CI_PREM</a>	Yes <a href="#">CI_PREM_K</a>	<a href="#">VAL-PREM</a>		CIPVPRMK <a href="#">Has dependencies</a>	CIPVPRMI
Characteristic	<a href="#">CI_PREM_CHAR</a>	No. The key is PREM_ID plus an edate and a char type.		CIPVPCHV		CIPVPCHI
Geo Loc	<a href="#">CI_PREM_GEO</a>	No. The key is PREM_ID plus geo loc type.		CIPVPGOV		CIPVPGOI
Alternate Address	<a href="#">CI_PRM_ALT_ADD R</a>	Yes <a href="#">CI_PRM_ALT_AD DR_K</a>		CIPVAPAV	CIPVAPAK <a href="#">Has dependencies</a>	CIPVAPAI

## Location Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

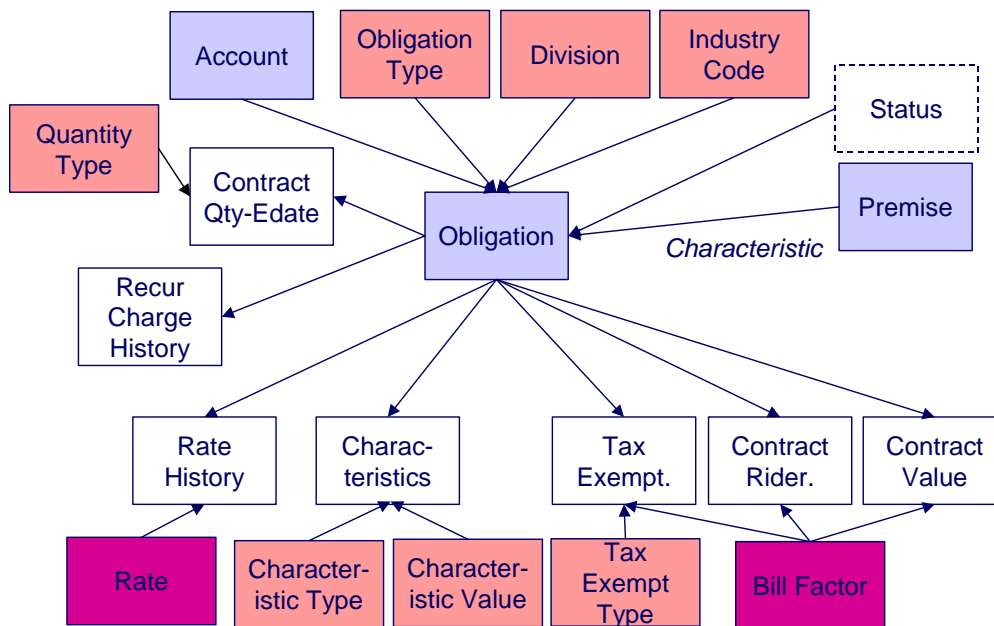
## Obligation

### Contents

- [Obligation Data Model](#)
- [Obligation Table Names](#)
- [Obligation Suggestions](#)

### Obligation Data Model

The following data model illustrates the obligation object.



### Obligation Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Obligation	<a href="#">CI_SA</a>	Yes <a href="#">CI_SA_K</a>	VAL-SA		CIPVSVAK <a href="#">Has dependencies</a>	CIPVSVAI
Characteristic	<a href="#">CI_SA_CHAR</a>	No. The key is obligation_ID plus an edate and a char type.		CIPVSACV		CIPVSACI

Contract Quantity Edate	<a href="#">CI_SA_CONT_QTY</a>	No. The key is obligation ID plus quantity type plus edate.		CIPVSAQV		CIPVSAQI
Message	<a href="#">CI_SA_MSG</a>	No. The key is obligation ID plus Bill message code.		CIPVSMGV		CIPVSMGI
Recurring Charge	<a href="#">CI_SA_RCHG_HIST</a>	No. The key is obligation ID plus edate.		CIPVSARV		CIPVSARI
Rate History	<a href="#">CI_SA_RS_HIST</a>	No. The key is obligation ID plus edate.		CIPVSAHV		CIPVSAHI
Tax Exempt	<a href="#">CI_SA_CONTERM</a> – this table is also used for the next 2 entities, the key contains CONTERM_TYPE_FLG that controls the entity	No. This key is obligation ID plus CONTERM_TYPE_FLG plus BF_CD plus START_DT		CIPVSAOV		CIPVSAOI
Contract Rider	<a href="#">CI_SA_CONTERM</a> – this table is also used for the previous and next entities, the key contains CONTERM_TYPE_FLG that controls the entity	No. This key is obligation ID plus CONTERM_TYPE_FLG plus BF_CD plus START_DT		CIPVSAOV		CIPVSAOI
Contract Value	<a href="#">CI_SA_CONTERM</a> – this table is also used for the previous 2 entities, the key contains CONTERM_TYPE_FLG that controls the entity	No. This key is obligation ID plus CONTERM_TYPE_FLG plus BF_CD plus START_DT		CIPVSAOV		CIPVSAOI

### Obligation Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

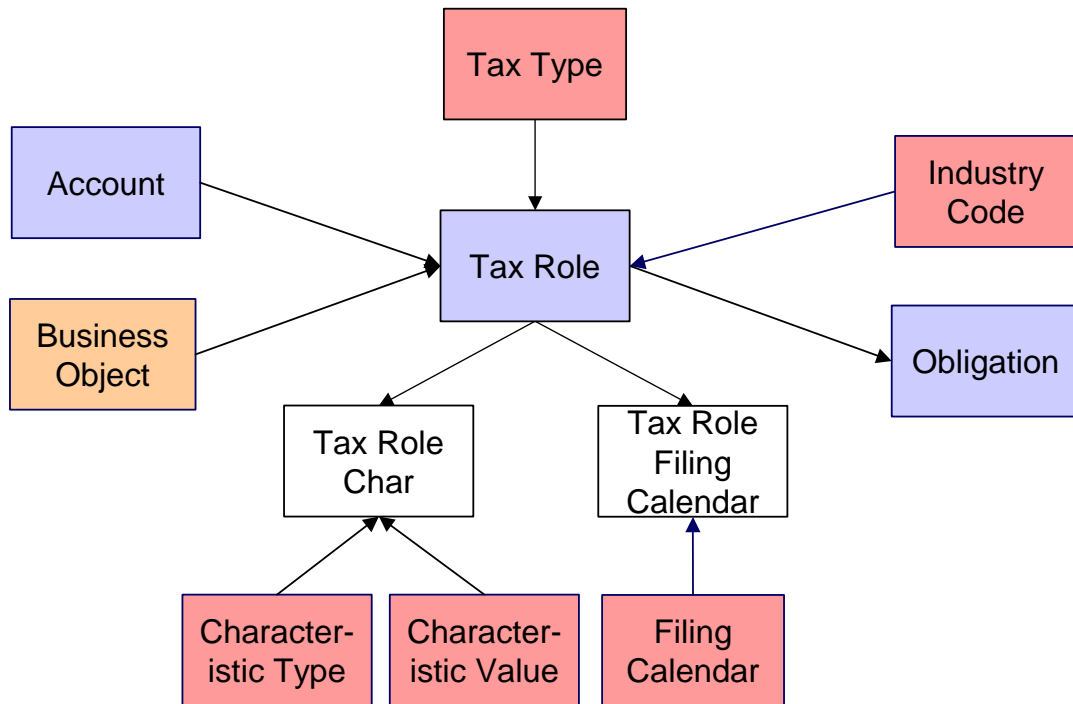
## Tax Role

### Contents

- [Tax Role Data Model](#)
- [Tax Role Table Names](#)
- [Tax Role Suggestions](#)

## Tax Role Data Model

The following data model illustrates the tax role object.



## Tax Role Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Tax Role	<a href="#">CI_TAX_ROLE</a>	Yes. <a href="#">CI_TAX_ROLE_K</a>	<a href="#">VAL-TAXR</a>	CIPVTXRV	CIPVTXRK	CIPVTXRI
Tax Role Filing Calendar	<a href="#">CI_TAX_ROLE_CAL</a>	No. The key is Tax Role ID and Effective Date.				
Tax Role Characteristic	<a href="#">CI_TAX_ROLE_CHAR</a>	No. The key is Tax Role ID, Characteristic Type, and Effective Date.		CIPVTXCV		CIPVTXCI

## Tax Role Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

## Transaction Data

This section describes the tables in which your transaction data (e.g., bills, payments, customer contacts, etc.) resides.

### Contents

- [Bill](#)
- [Payment](#)
- [Adjustment](#)
- [Customer Contact](#)
- [Rate Factor Value](#)
- [Payment Plans](#)
- [Billable Charge](#)
- [Collection Agency Referral](#)
- [Case](#)
- [Tax Form](#)
- [Collection Case](#)
- [Overpayment Process](#)

## Bill

### Contents

- [Bill Data Model](#)
- [Bill Table Names](#)
- [Bill Suggestions](#)

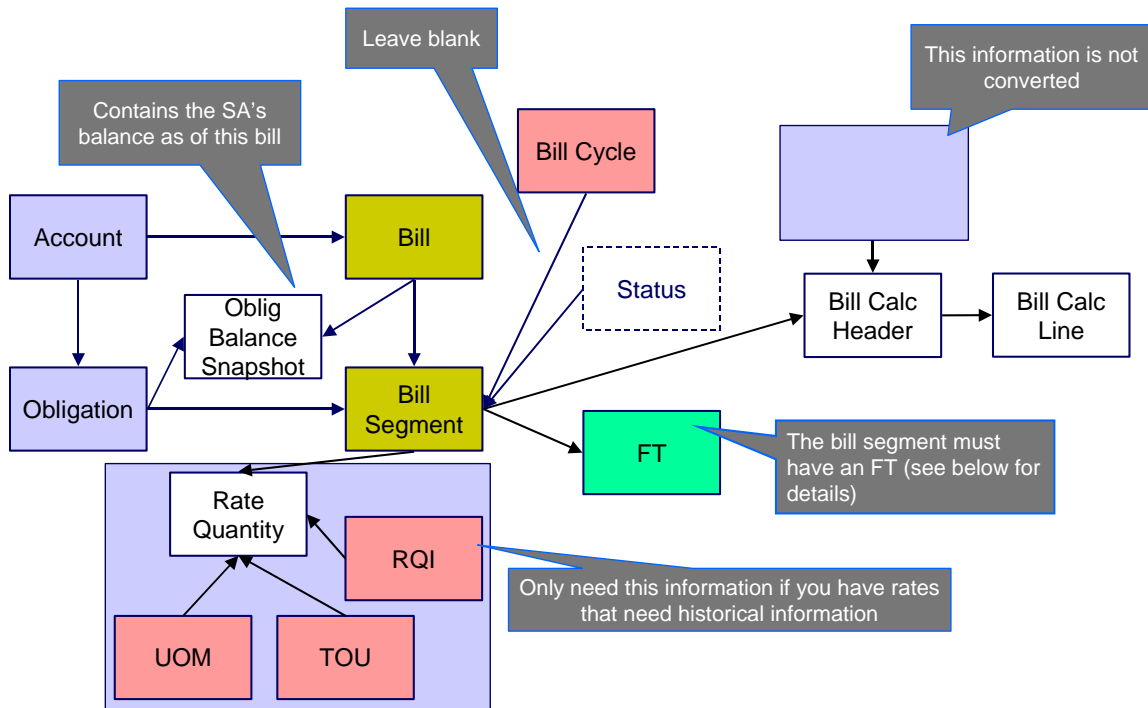
### Bill Data Model

### Contents

- [Bill - Main](#)
- [Bill - FT](#)
- [Bill Characteristics](#)
- [Bill Messages and Routing](#)

### Bill - Main

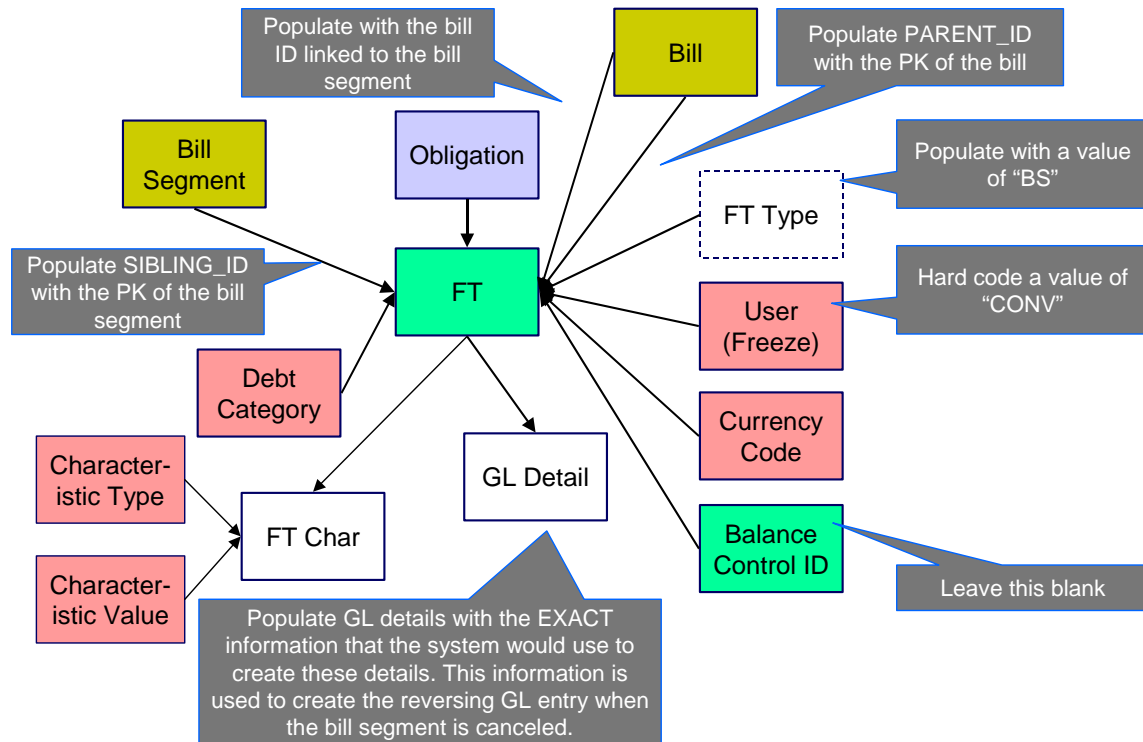
The following data model illustrates the bill object.



### Bill - FT

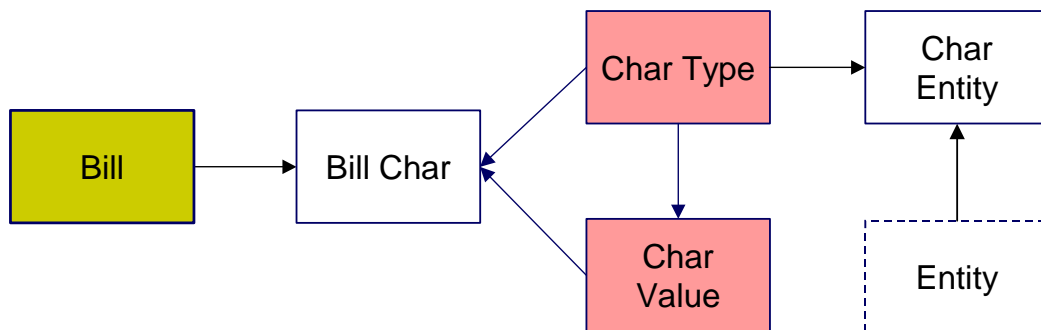
The following data model illustrates the FT that must be associated with a bill segment.





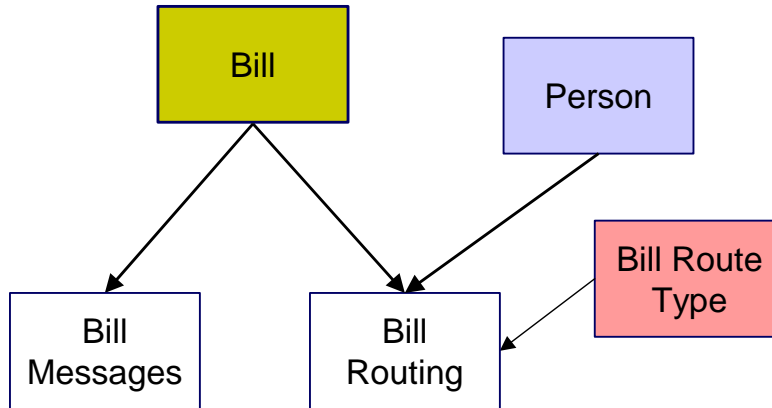
### Bill Characteristics

The following data model illustrates Bill Characteristics.



### Bill Messages and Routing

The following data model illustrates Bill Messages and Bill Routing.



## Bill Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Bill	<a href="#">CI BILL</a>	Yes <a href="#">CI BILL_K</a>	CIPVBLLV	CIPVBILK <a href="#">Has dependencies</a>	CIPVBLLI
Obligation Balance Snapshot	<a href="#">CI BILL_SA</a>	No. The key is bill ID plus obligation ID.	CIPVBSAV		CIPVBSAI
Bill Segment	<a href="#">CI BSEG</a>	Yes <a href="#">CI BSEG_K</a>	CIPVSEGV	CIPVBSGK <a href="#">Has dependencies</a>	CIPVSEGI
Calc Header	<a href="#">CI BSEG_CALC</a>	No. The key is bill segment id and a sequence number	CIPVBSCV		CIPVBSCI
Calc Lines	<a href="#">CI BSEG_CALC_L N</a>	No. The key is bill segment id, the header sequence number, and a sequence number	CIPVBSLV		CIPVBSLI
Read Detail	<a href="#">CI BSEG_READ</a>	No. The key is bill segment id and a sequence number.	CIPVSRRV		CIPVSRI
Asset Detail	<a href="#">CI BSEG_ITEM</a>	No. The key is bill segment id and a sequence number	CIPVBSIV		CIPVBSII
Rate Quantity	<a href="#">CI BSEG_SO</a>	No. The key is bill segment id, uom code, tou code and RQI code	CIPVSQTV		CIPVSQTI

FT (financial transaction)	<a href="#">CI_FT</a>	Yes <a href="#">CI_FT_K</a>	CIPVFTFV	CIPVFTXK <a href="#">Has dependencies</a>	CIPVFTFI
FT Characteristic	<a href="#">CI_FT_CHAR</a>	No. The key is FT id, char type code and a sequence number	CIPVFTCV		CIPVFTCI
FT GL (FT general ledger)	<a href="#">CI_FT_GL</a>	No. The key is FT id and a GL sequence number	CIPVFTGV		CIPVFTGI
Characteristics	<a href="#">CI_BILL_CHAR</a>	No. The key is bill id, char type code and a sequence number	CIPVBCHV		CIPVBCHI
Bill Messages	<a href="#">CI_BILL_MSGS</a>	No. The key is bill id and bill message code.	CIPVBLMV		CIPVBLMI
Bill Routing	<a href="#">CI_BILL_ROUTING</a>	No. The key is bill id and a sequence number	CIPVBLRV		CIPVBLRI

### Bill Suggestions

Most companies have found it impossible to load bill segment item, bill calc header and lines with sufficient information and therefore these tables are not populated. See the comments in the above ERD's for more information.

Please populate the columns on the FT that's associated with the bill segment as follows:

- CUR\_AMT should be set equal to the bill segment amount
- PAY\_AMT should be set equal to the bill segment amount
- CRE\_DTTM should be set equal to the bill segment end date / time
- FREEZE\_SW should be "Y"
- FREEZE\_DTTM should be set equal to the bill segment end date / time
- ARS\_DT should be set equal to the bill segment end date
- CORRECTION\_SW should be "N"
- REDUNDANT\_SW should be "N"
- NEW\_DEBIT\_SW should be "N"
- NOT\_IN\_ARS\_SW should be set to "N"
- SHOW\_ON\_BILL\_SW should be set to "N"
- ACCOUNTING\_DT should be set to the current date

- SCHED\_DISTRIB\_DT should be left blank
- CURRENCY\_CD should be the currency on the installation record
- BAL\_CTL\_GRP\_ID should be left blank
- XFERRED\_OUT\_SW should be set to "Y"
- PARENT\_ID should be set to the bill ID
- SIBLING\_ID should be set to the bill segment ID
- Do NOT create any GL details for the FT. If GL details are converted, ensure they are populated with the EXACT information the system would use to create them. This information is used to create the reversing GL entry when the bill segment is canceled.

## Payment

### Contents

- [Payment Data Model](#)
- [Payment Table Names](#)
- [Payment Suggestions](#)

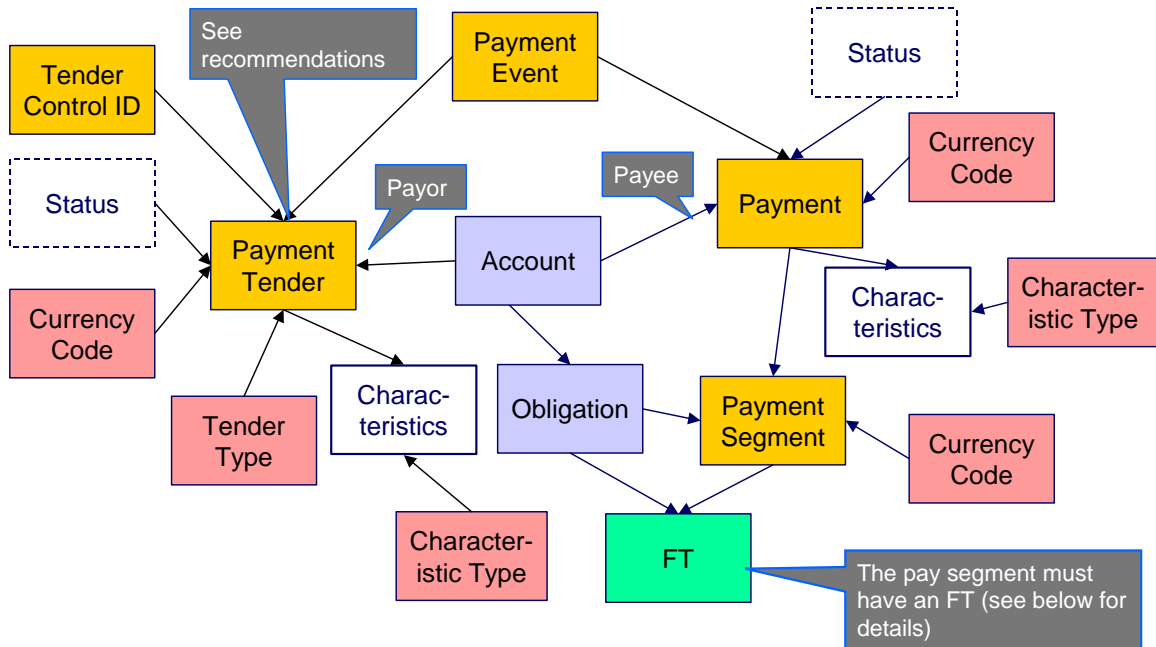
### Payment Data Model

#### Contents

- [Payment - Main](#)
- [Payment - FT](#)

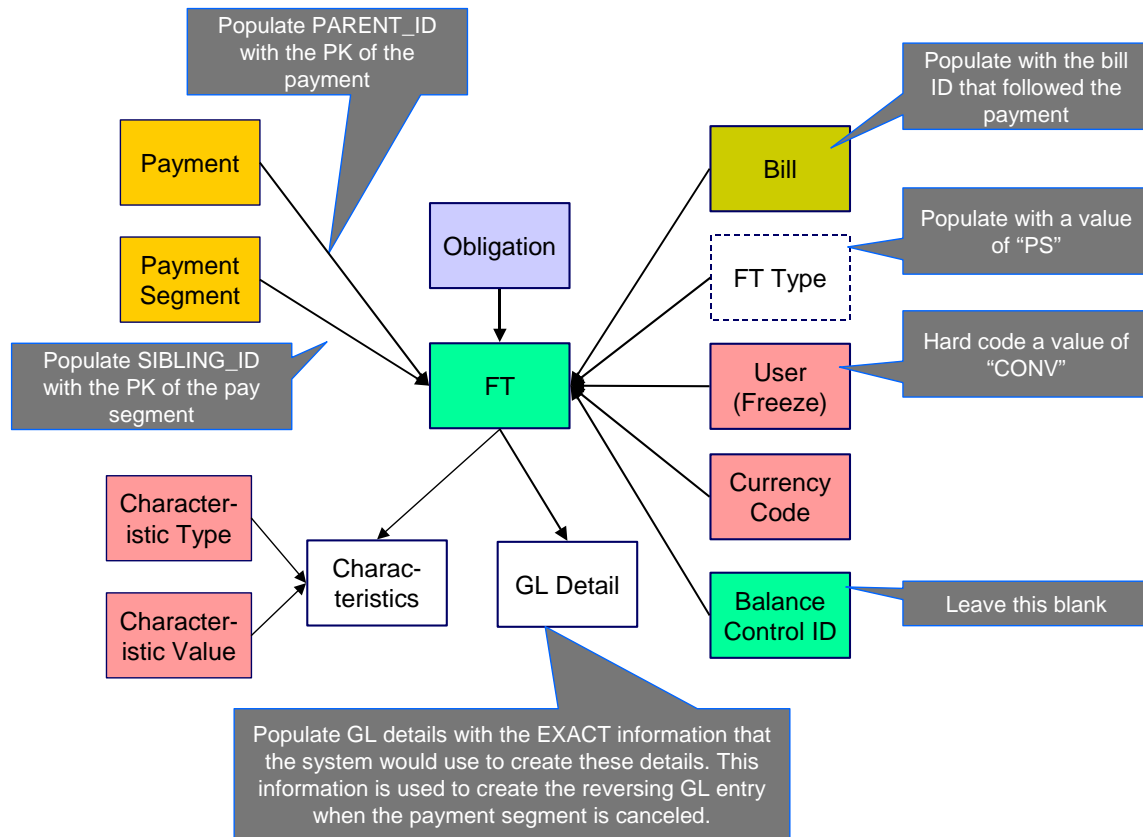
#### Payment - Main

The following data model illustrates the payment object.



### Payment - FT

The following data model illustrates the FT that must be associated with a payment segment.



### Payment Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Payment	<a href="#">CI_PAY</a>	Yes <a href="#">CI_PAY_K</a>	CIPVPAYV	CIPVPAYK <a href="#">Has dependencies</a>	CIPVPAYI
Payment Characteristic	<a href="#">CI_PAY_CHAR</a>	No. The key is PAY_ID, plus a sequence number and a char type	CIPVPYCV		CIPVPYCI
Payment Event	<a href="#">CI_PAY_EVENT</a>	Yes <a href="#">CI_PAY_EVENT_K</a>		CIPVPYEK <a href="#">Has dependencies</a>	CIPVPYEI
Payment Tender	<a href="#">CI_PAY_TNDR</a>	Yes <a href="#">CI_PAY_TNDR_K</a>	CIPVTNDV	CIPVTNDK <a href="#">Has dependencies</a>	CIPVTNDI
Payment Tender	<a href="#">CI_PAY_TNDR_CHAR</a>	No. The key is PAY_TENDER_ID,	CIPVTNCV		CIPVTNCI

Characteristic		plus a sequence number and a char type			
Payment Segment	<a href="#">CI_PAY_SEG</a>	Yes <a href="#">CI_PAY_SEG_K</a>	CIPVPSGV	CIPVPSGK <a href="#">Has dependencies</a>	CIPVPSGI
FT (financial transaction)	<a href="#">CI_FT</a>	Yes <a href="#">CI_FT_K</a>	CIPVFTFV	CIPVFTXK <a href="#">Has dependencies</a>	CIPVFTFI
FT Characteristic	<a href="#">CI_FT_CHAR</a>	No. The key is FT id, char type code and a sequence number	CIPVFTCV		CIPVFTCI
FT GL (FT general ledger)	<a href="#">CI_FT_GL</a>	No. The key is FT id and a GL sequence number	CIPVFTGV		CIPVFTGI

### Payment Suggestions

We recommend that you use the system to create a single deposit control and link to it a single tender control using the PRODUCTION tables. The tender control should reference a tender source of “conversion”. Use the prime key of the tender control as the foreign key on the tenders that you insert into the STAGING tables. This means you will have an invalid foreign key relationship on CI\_PAY\_TNDR (it will reference a tender control that doesn’t exist).

After converting the payments:

- Re-access the tender control in production and enter the appropriate amounts (per tender type) to balance the tender control.
- Re-access the deposit control in production and enter the appropriate amounts to balance the deposit control.

Please populate the columns on the FT that’s associated with the payment segment as follows:

- CUR\_AMT should be set equal to the payment segment amount
- PAY\_AMT should be set equal to the payment segment amount
- CRE\_DTTM should be set equal to the payment segment date / time
- FREEZE\_SW should be “Y”
- FREEZE\_DTTM should be set equal to the payment segment date / time
- ARS\_DT should be set equal to the payment segment date
- CORRECTION\_SW should be “N”
- REDUNDANT\_SW should be “N”
- NEW\_DEBIT\_SW should be “N”

- NOT\_IN\_ARS\_SW should be set to "N"
- SHOW\_ON\_BILL\_SW should be set to "N" on all payments other than payments that have been received since the last bill. For recent payments that you want to show on the next bill, this switch must be "Y"
- ACCOUNTING\_DT should be set to the current date
- SCHED\_DISTRIB\_DT should be left blank
- CURRENCY\_CD should be the currency on the installation record
- BAL\_CTL\_GRP\_ID should be left blank
- XFERRED\_OUT\_SW should be set to "Y"
- PARENT\_ID should be set to the payment ID
- SIBLING\_ID should be set to the payment segment ID
- Do NOT create any GL details for the FT. If GL details are converted, ensure they are populated with the EXACT information the system would use to create them. This information is used to create the reversing GL entry when the payment segment is canceled.

## Adjustment

### Contents

[Adjustment Data Model](#)  
[Adjustment Table Names](#)  
[Adjustment Suggestions](#)

### Adjustment Data Model

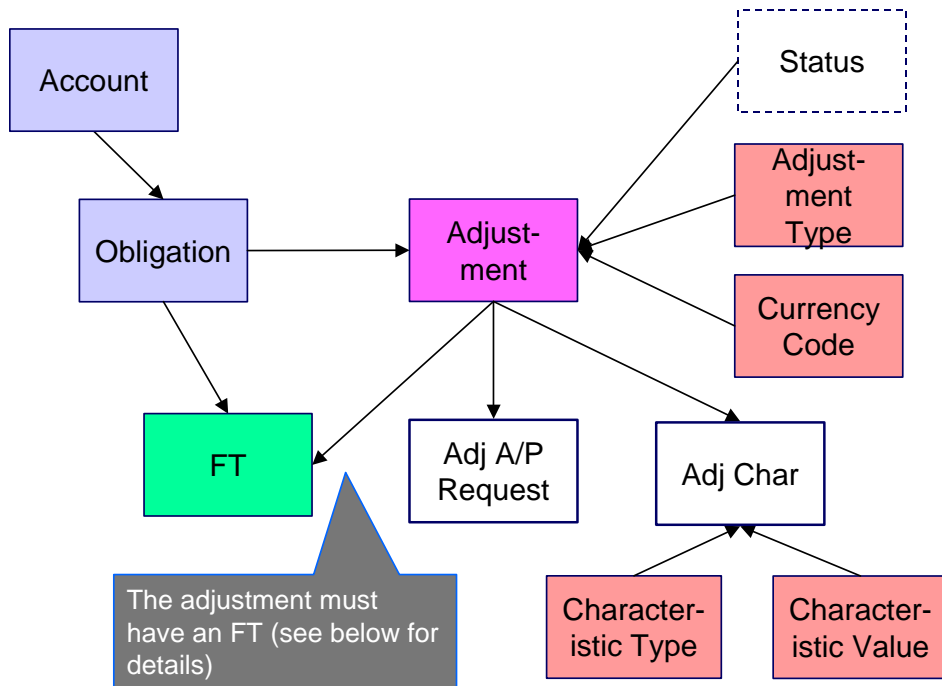
#### Contents

[Adjustment - Main](#)  
[Adjustment - FT](#)

#### [Adjustment - Main](#)

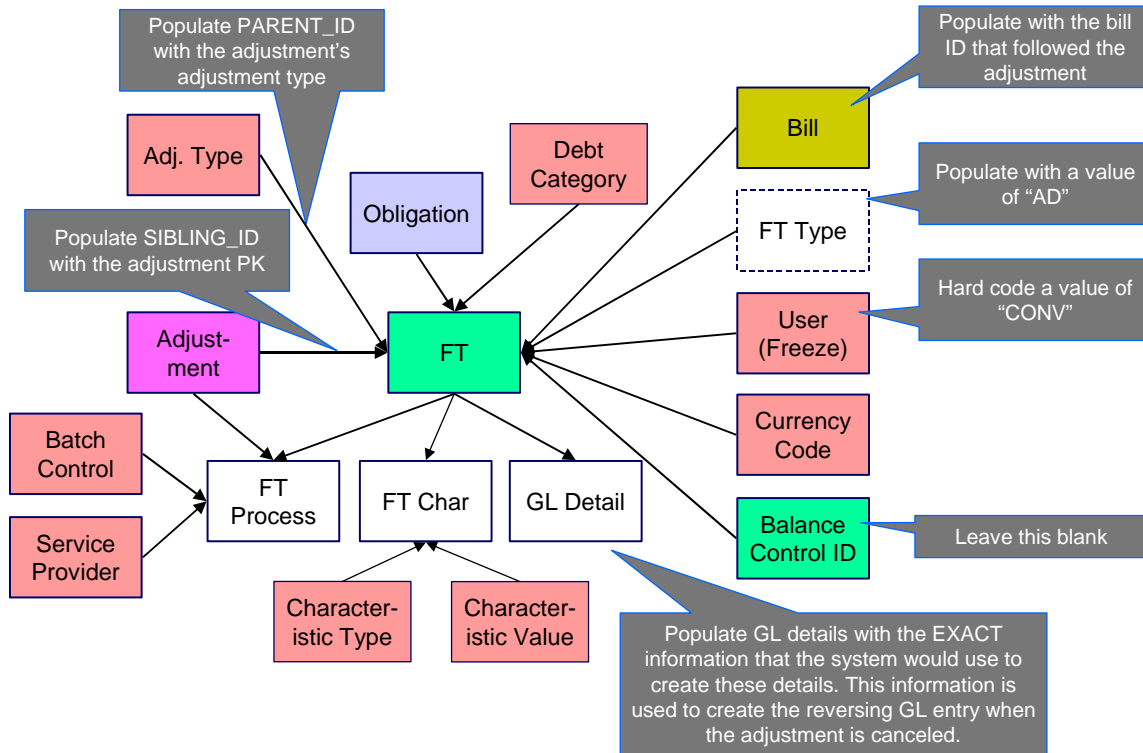
The following data model illustrates the adjustment object.





#### Adjustment - FT

The following data model illustrates the FT that must be associated with an adjustment segment.



## Adjustment Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Adjustment	<a href="#">CI_ADJ</a>	Yes <a href="#">CI_ADJ_K</a>	CIPVADJV	CIPVADJK <a href="#">Has dependencies</a>	CIPVADJI
Adjustment A/P Request	<a href="#">CI_ADJ_APREQ</a>	Yes <a href="#">CI_ADJ_APREQ_K</a>	CIPVAPRV	CIPVAPRK <a href="#">Has dependencies</a>	CIPVAPRI
FT (financial transaction)	<a href="#">CI_FT</a>	Yes <a href="#">CI_FT_K</a>	CIPVFTFV	CIPVFTXK <a href="#">Has dependencies</a>	CIPVFTFI
FT Characteristic	<a href="#">CI_FT_CHAR</a>	No. The key is FT id, char type code and a sequence number	CIPVFTCV		CIPVFTCI
FT GL (FT general ledger)	<a href="#">CI_FT_GL</a>	No. The key is FT id and a GL sequence number	CIPVFTGV		CIPVFTGI
FT Process	<a href="#">CI_FT_PROC</a>	No. The key is FT	CIPVFTPV		CIPVFTPI

		id and a sequence number			
Adjustment Characteristic	<a href="#">CI_ADJ_CHAR</a>	No. The key is adjustment id, char type code and a sequence number	CIPVADCV		CIPVADCI

### Adjustment Suggestions

Please populate the columns on the FT that's associated with the adjustment as follows:

- CUR\_AMT should be set equal to the adjustment amount
- PAY\_AMT should be set equal to the adjustment amount
- CRE\_DTTM should be set equal to the adjustment date / time
- FREEZE\_SW should be "Y"
- FREEZE\_DTTM should be set equal to the adjustment date / time
- ARS\_DT should be set equal to the adjustment date
- CORRECTION\_SW should be "N"
- REDUNDANT\_SW should be "N"
- NEW\_DEBIT\_SW should be "N"
- NOT\_IN\_ARS\_SW should be set to "N"
- SHOW\_ON\_BILL\_SW should be set to "N" on all adjustments other than adjustments that have been generated since the last bill. For recent adjustments that you want to show on the next bill, this switch must be "Y"
- ACCOUNTING\_DT should be set to the current date
- SCHED\_DISTRIB\_DT should be left blank
- CURRENCY\_CD should be the currency on the installation record
- BAL\_CTL\_GRP\_ID should be left blank
- XFERRED\_OUT\_SW should be set to "Y"
- PARENT\_ID should be set to the adjustment's adjustment type
- SIBLING\_ID should be set to the adjustment ID
- Do NOT create any GL details for the FT. If GL details are converted, ensure they are populated with the EXACT information the system would use to create them. This information is used to create the reversing GL entry when the adjustment is canceled.

## Customer Contact

### Contents

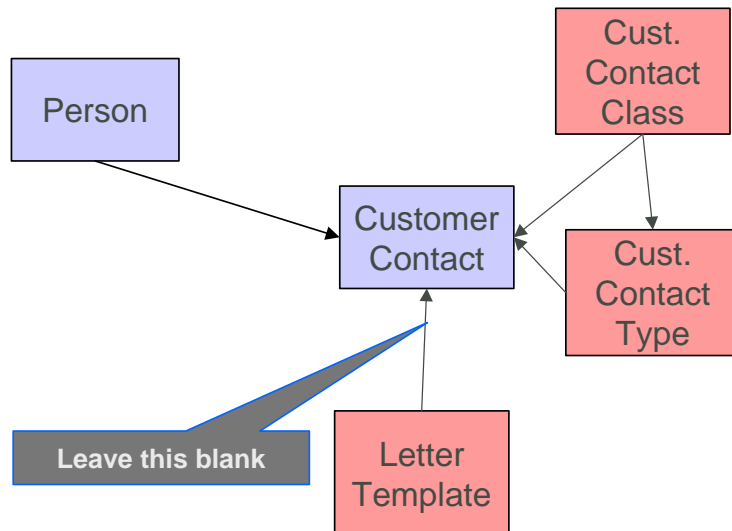
[Customer Contact Data Model](#)

## Customer Contact Table Names

### Customer Contact Suggestions

#### Customer Contact Data Model

The following data model illustrates the Customer Contact object.



#### Customer Contact Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Customer Contact	<a href="#">CI_CC</a>	Yes <a href="#">CI_CC_K</a>	CIPVCSCV	CIPVCCTK <a href="#">Has dependencies</a>	CIPVCSCI

#### Customer Contact Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

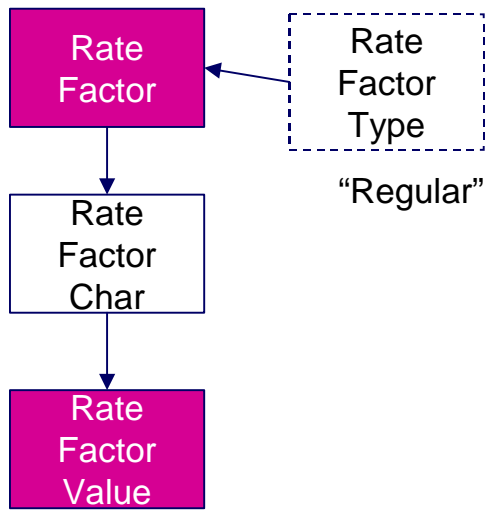
## Rate Factor Value

### Contents

- [Rate Factor Value Data Model](#)
- [Rate Factor Value Table Names](#)
- [Rate Factor Value Suggestions](#)

### Rate Factor Value Data Model

The following data model illustrates the rate factor objects.



### Rate Factor Value Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Rate Factor Value	<a href="#">CI_BF_VAL</a>	No. The key is BF_CD plus CHAR_TYPE_CD plus CHAR_VAL plus EFFDT	CIPVBFVV		CIPVBFVI (Not threadable)

### Rate Factor Value Suggestions

N/A

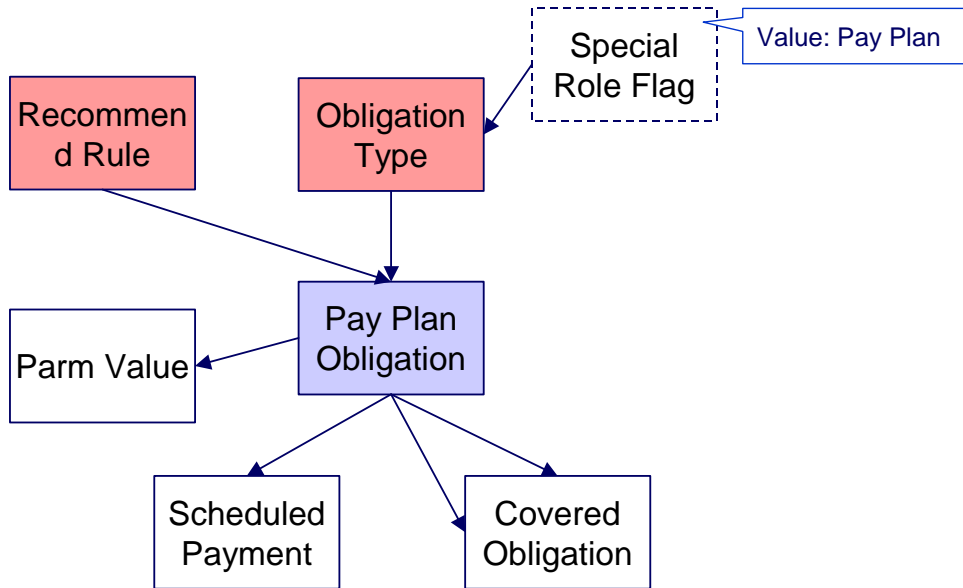
## Payment Plans

### Contents

- [Payment Plan Data Model](#)
- [Payment Plan Table Names](#)
- [Payment Plan Suggestions](#)

### Payment Plan Data Model

The following data model illustrates the Payment Plan objects.



#### Payment Plan Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Covered Obligation	<a href="#">CI_NB_SA</a>	No. The key is SA_ID plus CVRD_SA_ID	CIPVNBSV		CIPVNBSI
Scheduled Payments	<a href="#">CI_NB_SCHED_PAY</a>	Yes <a href="#">CI_NB_SCHED_PAY_K</a>	CIPVNSPV	CIPVNSPK <a href="#">Has dependencies</a>	CIPVNSPI
Pay Plan Parameters	<a href="#">CI_SA_NB_PARM</a>	No. The key is SA_ID plus Sequence Number.	CIPVNPMV		CIPVNPMI

#### Payment Plan Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

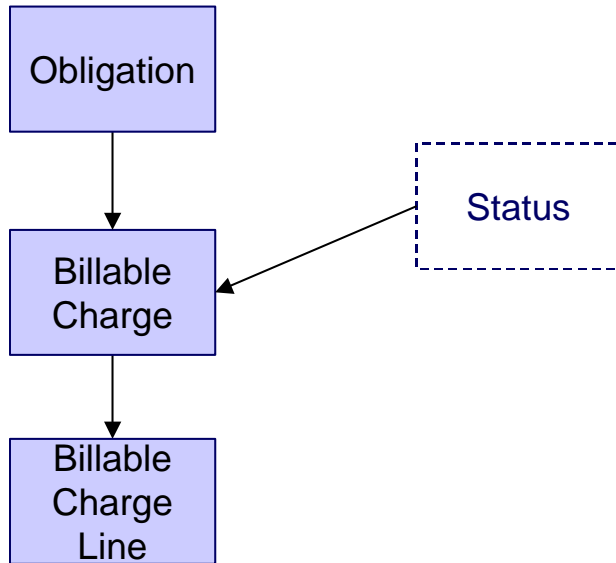
## Billable Charge

### Contents

- [Billable Charge Data Model](#)
- [Billable Charge Table Names](#)
- [Billable Charge Suggestions](#)

### Billable Charge Data Model

The following data model illustrates the Billable Charge objects.



### Billable Charge Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Billable Charge	<a href="#">CI BILL CHG</a>	Yes. <a href="#">CI BILL CHG K</a>	<a href="#">VAL-BCHG</a>	CIPVBCGV	CIPVBCGK	CIPVBCGI
Billable Charge Line	<a href="#">CI B CHG LINE</a>	No. The key is billable charge id and a sequence number		CIPVBCLV		CIPVBCLI

### Billable Charge Suggestions

N/A

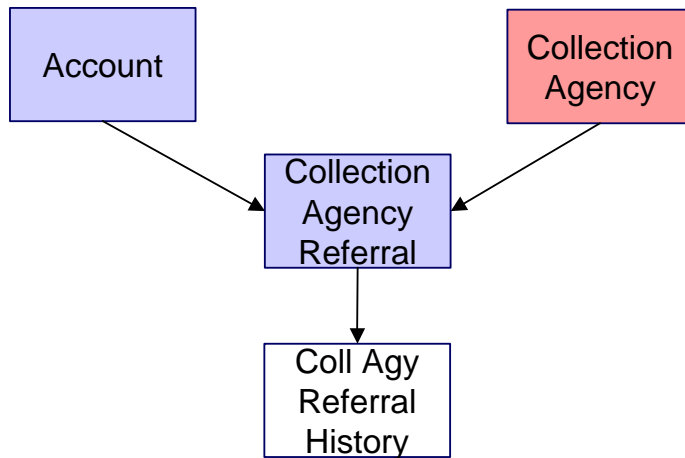
## Collection Agency Referral

### Contents

- [Collection Agency Referral Data Model](#)
- [Collection Agency Referral Table Names](#)
- [Collection Agency Referral Suggestions](#)

### Collection Agency Referral Data Model

The following data model illustrates the Collection Agency Referral object.



### Collection Agency Referral Table Names

Data Model Name	Table Name	Generated Keys	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Collection Agency Referral	<a href="#">CI_COLL_AGY_REF</a>	Yes. <a href="#">CI_COLL_AGY_REF_K</a>	CIPVCARV	CIPVCARK	CIPVCARI
Collection Agency Referral History	<a href="#">CI_COLL_AGY_HIS</a>	No. The key is collection agency referral id and characteristic type code	CIPVARHV		CIPVARHI

### Collection Agency Referral Suggestions

N/A

## Case

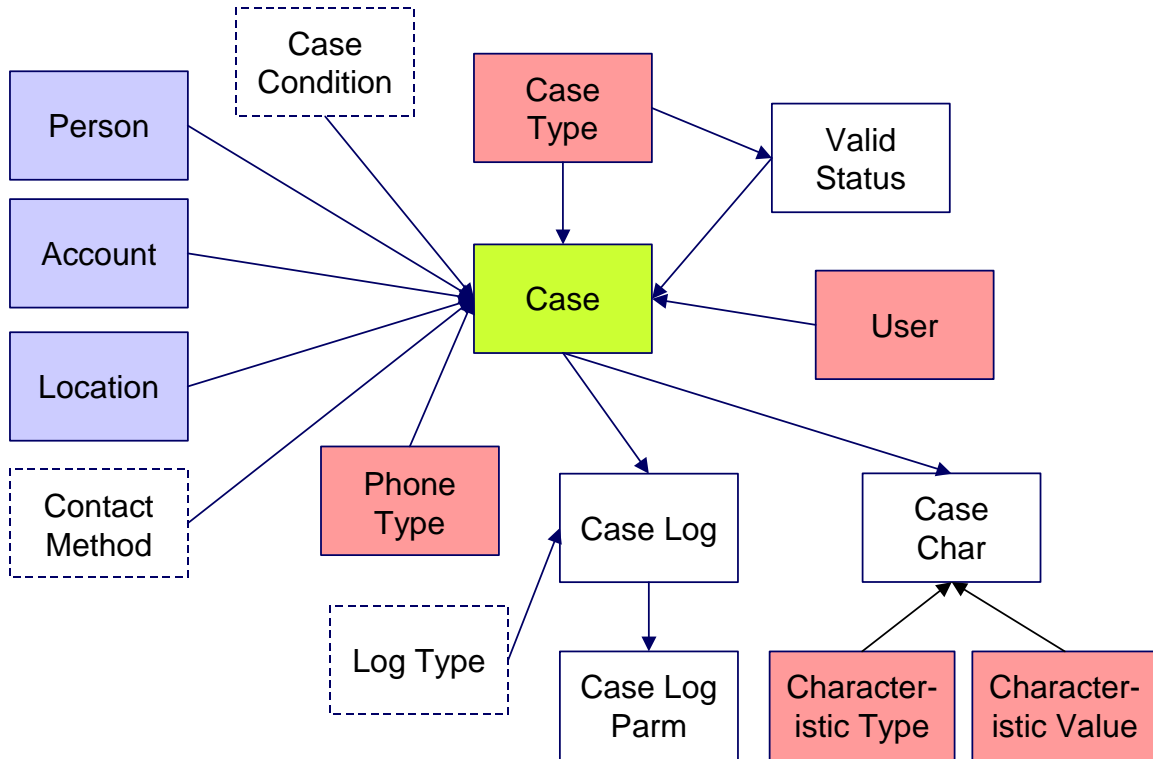
### Contents

- [Case Data Model](#)
- [Case Table Names](#)
- [Case Suggestions](#)

### Case Data Model

The following data model illustrates the Case object.





## Case Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Case	<a href="#">CI_CASE</a>	Yes. <a href="#">CI_CASE_K</a>	<a href="#">VAL-CASE</a>		CIPVCSEK	CIPVCSEI
Case Characteristic	<a href="#">CI_CASE_CHAR</a>	No. The key is case id, char type code and a sequence number		CIPVCCHV		CIPVCCHI
Case Log	<a href="#">CI_CASE_LOG</a>	No. The key is case id and a log sequence number		CIPVCLGV		CIPVCLGI
Case Log Parameter	<a href="#">CI_CASE_LOG_PARAMETER</a>	No. The key is case id, log sequence number and a parameter sequence number		CIPVCPAV		CIPVCPAI

## Case Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

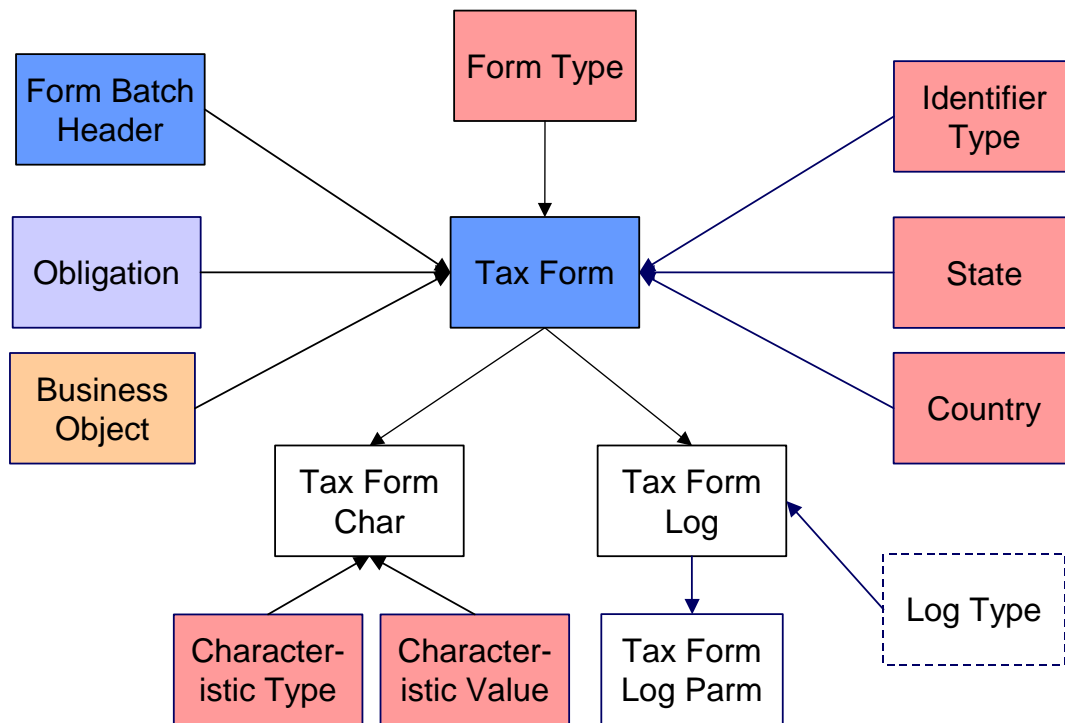
## Tax Form

### Contents

- [Tax Form Data Model](#)
- [Tax Form Table Names](#)
- [Tax Form Suggestions](#)

### Tax Form Data Model

The following data model illustrates the Tax Form object.



### Tax Form Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Tax Form	<a href="#">CI_TAX_FORM</a>	Yes. <a href="#">CI_TAX_FORM_K</a>	<a href="#">VAL-TXFR</a>	CIPVTXFV	CIPVTXFK	CIPVTXFI

Tax Form Characteristic	<a href="#">CI_TAX_FORM_CHARACTER</a>	No. The key is Tax Form ID, Characteristic Type, and Sequence.		CIPVTFCV		CIPVTFCI
Tax Form Log	<a href="#">CI_TAX_FORM_LOG</a>	No. The key is Tax Form ID and Sequence.		CIPVTFLV		CIPVTFLI
Tax Form Log Parameter	<a href="#">CI_TAX_FORM_LOG_PARAMETER</a>	No. The key is Tax Form ID, Sequence, and Message Parameter Sequence.		CIPVTLPV		CIPVTLPI

### Tax Form Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

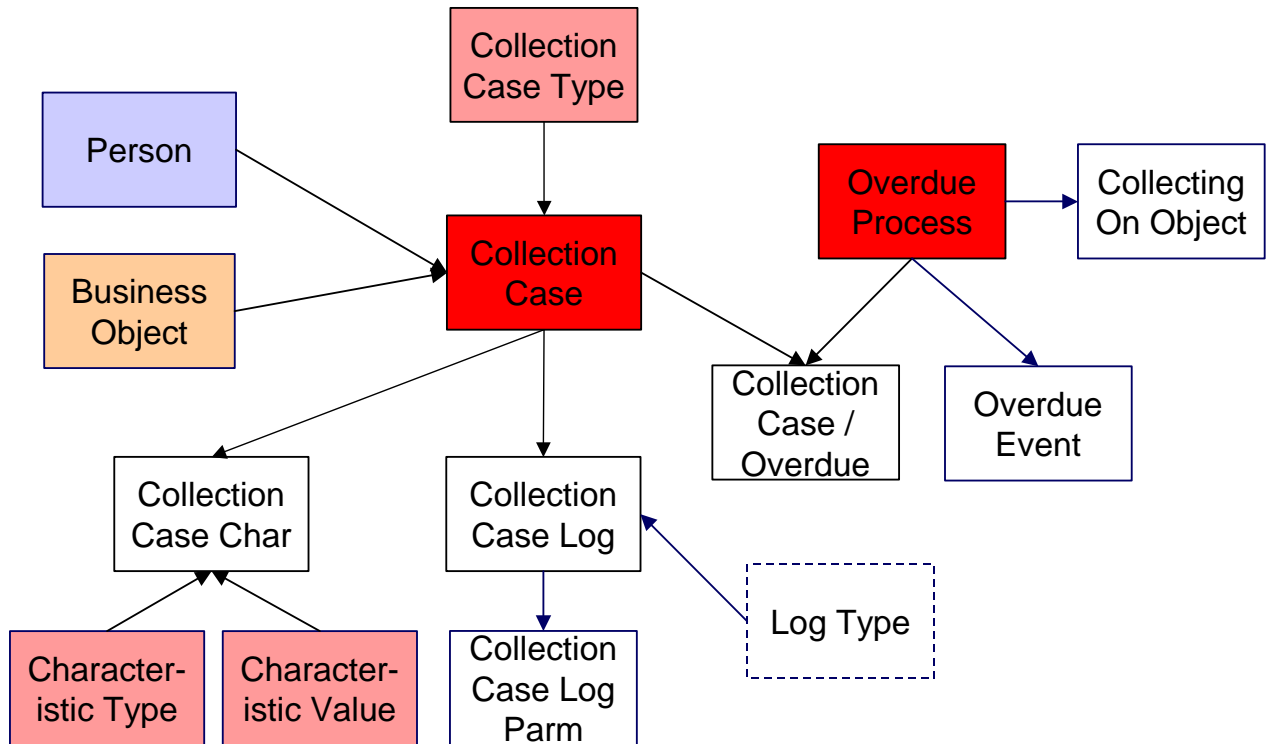
## Collection Case

### Contents

- [Collection Case Data Model](#)
- [Collection Case Table Names](#)
- [Collection Case Suggestions](#)

### Collection Case Data Model

The following data model illustrates the Collection Case object.



Collection Case Table Names

Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Collection Case	<a href="#">CI_COLL_CASE</a>	Yes. <a href="#">CI_COLL_CASE_K</a>	<a href="#">VAL-CLCS</a>	CIPVCCSV	CIPVCCSK	CIPVCCSI
Collection Case Characteristic	<a href="#">CI_COLL_CASE_CHARACTER</a>	No. The key is Collection Case ID. Characteristic Type, and Sequence.		CIPVCCCV		CIPVCCCI
Collection Case Log	<a href="#">CI_COLL_CASE_LOG</a>	No. The key is Collection Case ID and Sequence.		CIPVCCLV		CIPVCCLI

Collection Case Log Parameter	<a href="#">CI_COLL_CASE_LOG_PARM</a>	No. The key is Collection Case ID, Sequence, and Message Parameter Sequence.		CIPVCCPV		CIPVCCPI
Collection Case Overdue Process	<a href="#">CI_COLL_CASE_OD</a>	No. The key is Collection Case ID and Overdue Process ID.		CCIPVCCOV		CCIPVCCOI

### Collection Case Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

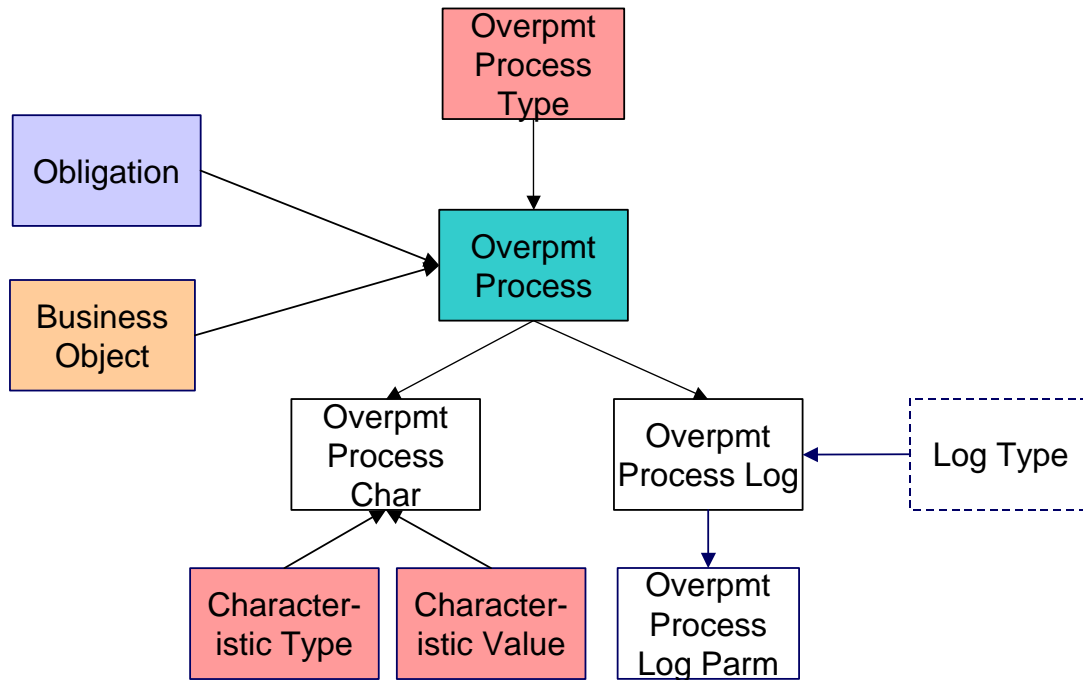
## Overpayment Process

### Contents

- [Overpayment Process Data Model](#)
- [Overpayment Process Table Names](#)
- [Overpayment Process Suggestions](#)

### Overpayment Process Data Model

The following data model illustrates the Overpayment Process object.



## Overpayment Process Table Names

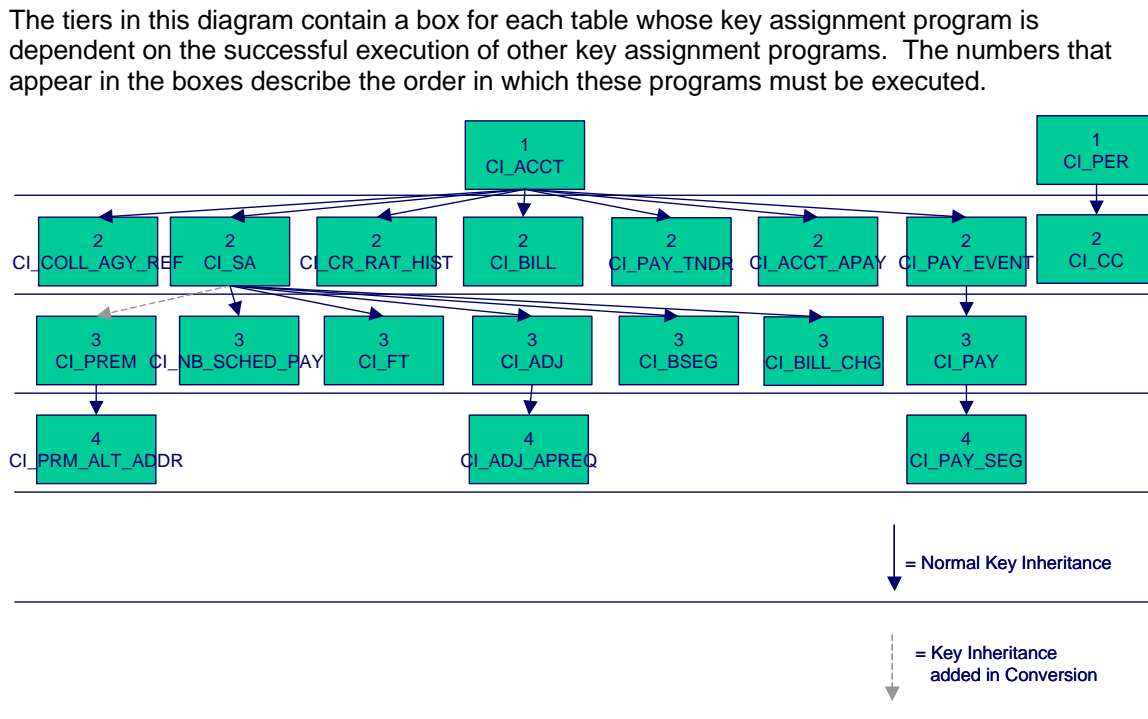
Data Model Name	Table Name	Generated Keys	Object Validation Batch Control	Referential Integrity Validation Batch Control	Key Assignment Batch Control	Insertion Batch Control
Overpayment Process	<a href="#">CI_OP_PROC</a>	Yes. <a href="#">CI_OP_PROC_K</a>	<a href="#">VAL-OVPY</a>	CIPVOPPV	CIPVOPPK	CIPVOPPI
Overpayment Process Character	<a href="#">CI_OP_PROC_CHAR</a>	No. The key is Overpayment Process ID, Characteristic Type, and Sequence.		CIPVOPCV		CIPVOPCI
Overpayment Process Log	<a href="#">CI_OP_PROC_LOG</a>	No. The key is Overpayment Process ID and Sequence.		CIPVOPLV		CIPVOPLI
Overpayment Process Log Parameter	<a href="#">CI_OP_PROC_LOG_PARM</a>	No. The key is Overpayment Process ID, Sequence, and Message Parameter Sequence.		CIPVOPMI		CIPVOPMI

## Overpayment Process Suggestions

This maintenance object includes a character large object field that your organization may be using to capture implementation specific data as defined by your business objects. For records of this type, the process to insert the records to the staging table is responsible for populating the data in this CLOB as per the record's business object schema.

## Program Dependencies

The programs used to assign production keys are listed under [Master Data](#) and [Transaction Data](#) (in the Table Names matrices). Most of these programs have no dependencies (i.e., they can be executed in any order you please). The only exceptions to this statement are illustrated in the following diagram.

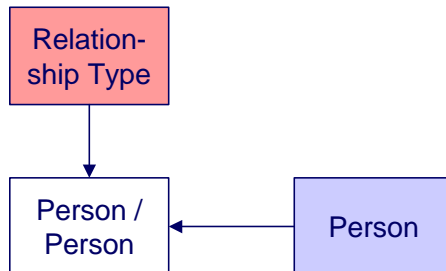


Please refer to the various "Table Names" sections above for the respective names of the programs to allocate each table's keys.

**Take note!** Prior to running the key generation program for a particular object, it is required that any previously generated keys be cleared from the key allocation tables and the key allocation temporary storage table. It is recommended that the key allocation tables be analyzed between runs to maximize performance.

## Appendix A - Entity Relationship Diagramming Standards

Because all data is stored in relational table, you need to be able to read diagrams that illustrate relationships between the various tables. The following entity diagram uses every diagramming notation used in this course:



### Contents

Entity  
Color Coding  
Relationships

## Entity

Every box on the above diagram represents an entity (i.e., a table). An entity may be a physical entity, such as a Person, or a logical construct, such as an Account.



## Color Coding

If you can view this document in color, you will notice that each entity is colored. The color indicates the “subsystem” which governs the entity. Know the governing subsystem is important because:









- The system’s menu structure is subsystem-oriented (i.e., if you know the subsystem, you will know how to use the menus to navigate to the page used to view and update the entity).
- The system’s documentation is subsystem-oriented (i.e., if you know the subsystem, you will know which chapter contains information about the entity).

Some entities are not color-coded (i.e., they are white). These entities do not have a dedicated page, as they are part of a parent entity. For example, the Person / Person entity above is related to the Person object and does not have its own page. You must display the parent entity in order to view such an entity. For example, if you want to look at Person / Person information, you must go to the Person page.

The following table describes the colors utilized in the documentation:

Color	Subsystem
	Taxpayer Information
	Admin (Control) Table. These tables are referenced as foreign keys on master and transaction tables. We do not document the names of these tables in this document as the table names are easily accessible using the Table transaction.



	N/A – the entity is maintained in respect of a higher level entity.
	N/A – the values in these types of entities are defined in a special table referred to as the lookup table. In order to determine the valid values for a column that references a lookup table, use the name of the column as the search value on the Look Up user interface.
	Rates
	Billing
	Financial Transaction
	Payment
	Adjustment
	Case

## Relationships

The solid line connecting the two entities that is terminated by an arrow represents a relationship between two entities. You read the relationship from the entity without the arrow to the entity with the arrow. For example, the line between account type and Account illustrates that an account type may have many Accounts, but an Account may be part of a single account type.

## Appendix B - Multiple Owners In A Single Database

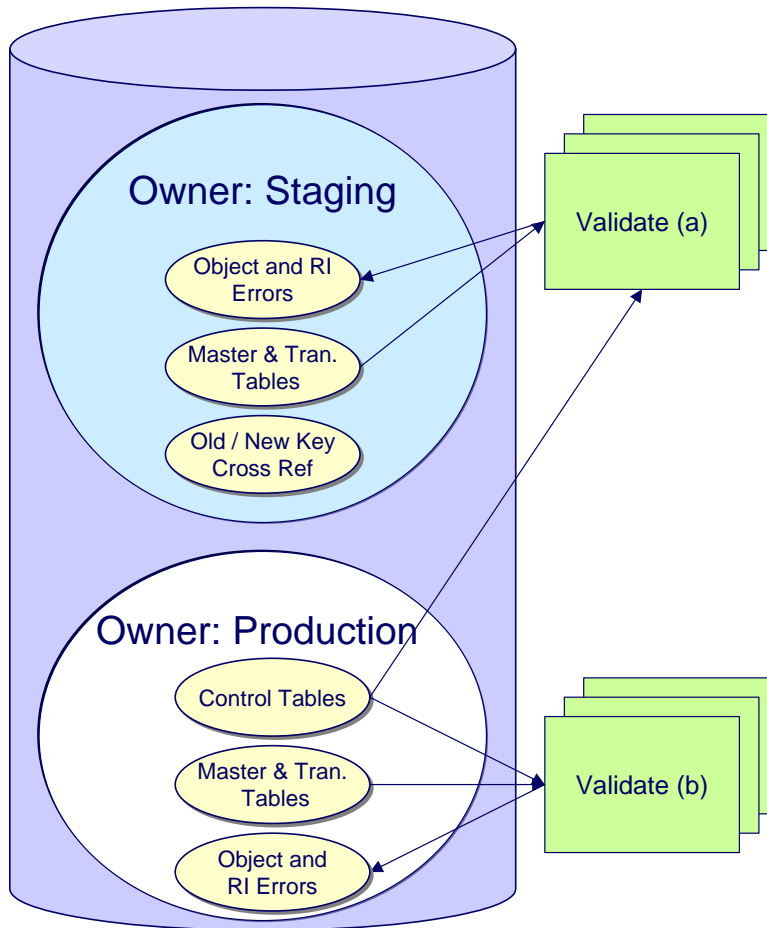
In the schematic referenced in the [Introduction](#), you'll notice that there are two table owners in the system database. We refer to the first owner as "staging" and the second owner as "production".

The staging owner is linked to the tables into which you insert your pre-validated data. These tables have an owner ID of **CISSTG**.

**Multiple staging databases.** It is possible to have multiple staging databases. In this situation, each one would have a unique owner ID, e.g., **CISSTG1**, **CISSTG2**, etc.

The production owner is linked to the tables used by your production system. These tables have an owner ID of **CISADM**.

When the validation programs run against your staging data, they validate the staging data against the production control tables (and insert errors into the staging error table). This means that the SQL statements that access / update master and transaction data need to use the staging owner (**CISSTG**). Whereas the SQL statements that access control tables need to use the production owner (**CISADM**).



But notice that when these same programs run against production (Validate (b)), the SQL statements will never access the staging owner. Rather, they all point at the production owner.

This is accomplished as follows:

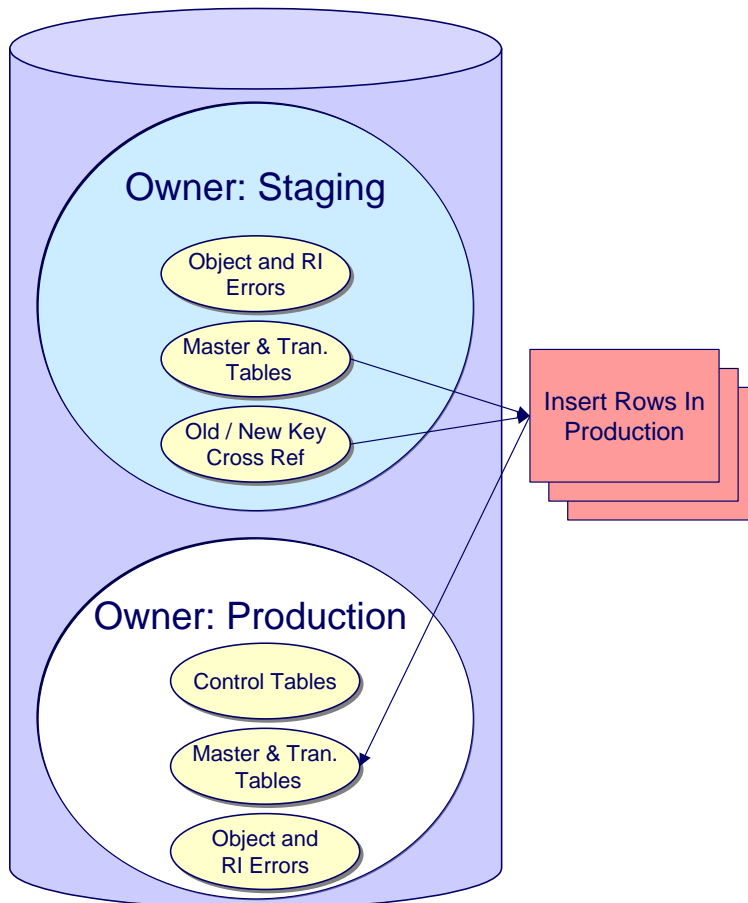
- A separate application server must exist for each owner. Each application server points at a specific Tuxedo server. The Tuxedo server references a specific database user ID.
- The database user ID associated with the staging database uses **CISSTG** as the owner for the master and transaction tables, but it uses **CISADM** as the owner of the production control tables.
- The database user ID associated with the production database uses **CISADM** as the owner for the master, transaction, and control tables.

You may wonder why we went to this trouble. There are several reasons:

- We wanted to re-use the validation logic that exists in the programs that validate your production data. In order to achieve this, these programs must sometimes point at the staging owner, and other times they must point at the production owner (and this must be transparent to the programs otherwise two sets of SQL would be necessary).
- We wanted to let you use the application to look at and correct staging data. This can be accomplished by creating an application server that points at your staging database with the ownership characteristics described above.

- We wanted the validation programs to be able to validate your production data (in addition to your staging data). Why would you want to validate production data if only clean data can be added to production? Consider the following scenarios:
  - After an upgrade, you might want to validate your production data to ensure your pre-existing user-exit logic still works.
  - You may want to conduct an experiment of the ramifications of changing your validation logic. To do this, you could make a temporary change to user exit logic (in production) and then run the validation jobs on a random sample basis.
  - You forget to run a validation program before populating production and you want to see the damage. If you follow the instructions in this document, this should never happen. However, accidents happen. And if they do, at least there's a way to determine the ramifications.

While the redirection of owner ID's is a useful technique for the validation programs, it cannot be used by the key assignment and production insert programs? Why, because these programs have to access the same tables but with different owners. For example, the program that inserts rows into the person table must select rows from staging.Person and insert them into production.Person.



This is accomplished as follows:

- Views exist for each table that exists in both databases. These views have hard-coded the database owner **CISADM** (production). For example, there is a view called CX\_PER that points at person table in production.
- The key assignment and insertion programs use these views whenever then need to access production data.

## Appendix C - Known Oddities

---

Be aware that the following tables reference master data (e.g., persons, accounts). This means that if you look at them using a user ID that defaults ownership to the staging level, you will not be able to see the related master data (because the person / account doesn't exist in the staging owner's tables).

- Collection Agency. References a person.
- Tender Source. References a suspense account.

# Oracle Enterprise Taxation Management

## 2.2.0 Release Contents

This document describes new / enhanced features in release 2.2.0 for Oracle Enterprise Taxation Management.

Refer to [Framework 2.2.0 Release Contents](#) for information on new / enhanced features in release 2.2.0 for Framework.

Refer to [Framework 2.2.0 Release Notes Addendum](#) for information on enhancements added to Framework since the release of version 2.2.0.

### Contents

- [Taxpayer Information Enhancements](#)
- [Payment Enhancements](#)
- [Financial Enhancements](#)
- [Billing Enhancements](#)
- [Adjustment Enhancements](#)
- [Collections](#)
- [Pay Plan Enhancements](#)
- [Support for Overpayment Processing](#)
- [Forms Processing](#)
- [Miscellaneous Enhancements](#)
- [Asset Functionality Deprecation](#)
- [New Application Services](#)
- [Changed Application Services](#)
- [Application Services No Longer Applicable](#)
- [User Exit Deprecation](#)
- [Planned Background Process Deprecation](#)

## Taxpayer Information Enhancements

---

### Contents

- [Enhanced Letter Creation](#)
- [Person Enhancements](#)
- [Account Enhancements](#)
- [Customer Contact Enhancements](#)
- [Tax Type Enhancements](#)
- [Tax Role Introduced](#)
- [Obligation Enhancements](#)
- [Premise Renamed Location](#)
- [Location Enhancements](#)

## Enhanced Letter Creation

In addition to supporting DOC1 software, the Letter Print functionality now supports creating letters online using Oracle BI Publisher. You can display a letter online associated with a customer contact by clicking **Display Letter** on the Customer Contact page.

**Note.** Creating letters in batch using this mechanism is not supported in this release. You can still create letters in batch using DOC1.

## Person Enhancements

[Person](#) has been enhanced to reference a new configuration table [Person Type](#). The person/business indicator on Person has been moved to the person type.

**Upgrade.** Person types **P** and **B** are provided when upgrading to this release. Any person that referenced the person/business indicator of **P** is upgraded to reference the new person type **P**. Any person that referenced the person/business indicator of **B** is upgraded to reference the new person type **B**.

Person has been enhanced to include a CLOB field to allow for additional site-specific elements to be added. In order to define the elements for this field, a person Business Object may be defined on the associated person type.

The seasonal address collection on the [Person - Correspondence](#) page has been changed to a Miscellaneous Address collection. A new address type lookup field is introduced to allow an implementation to categorize the address. A value of **Seasonal** is provided as a value for this address type field and the season boundary fields (MM/DD - MM/DD) are only available if the address type is **Seasonal**.

**Upgrade.** Any existing address in the collection is updated to reference an address type of **Seasonal**.

## Account Enhancements

In this release, [account](#) has been enhanced to include a CLOB field to allow for additional site-specific elements to be added. In order to define the elements for this field, an account Business Object may be defined on the associated [account type](#).

## Customer Contact Enhancements

In this release, [customer contact](#) has been enhanced to include a CLOB field to allow for additional site-specific elements to be added. In order to define the elements for this field, a customer contact Business Object may be defined on the associated [customer contact type](#).

## Tax Type Enhancements

In this release the [tax type](#) has been converted to a "new style" maintenance object with a business object used to define its behavior. The user interface has changed to a portal / zone user interface.

A CLOB has been added along with a field to configure the [tax role](#) applicability for this tax type.

**Upgrade.** The system provides a base business object for tax type that includes elements for the fields that existed on the tax type in prior releases. In addition the tax role applicability field is included and all existing tax types are updated to indicate tax roles are ***Not Applicable***.

## Tax Role Introduced

In this release tax role is introduced. Refer to [Tax Roles](#) for more information.

**Base business object.** The base product provides a business object for a filing period based tax role and handles effective dated changes to the tax role's filing calendar. Refer to the business object metadata for more information.

## Obligation Enhancements

Tax role has been added to Obligation and is required or not allowed based on the obligation type's tax type's configuration.

Filing period has been added to Obligation and is required or not allowed based on the [obligation type](#)'s configuration.

Override filing and payment due dates have been added to obligation. Refer to [Due Dates for Filing Period Based Obligations](#) for more information.

In this release, [obligation](#) has been enhanced to include a CLOB field to allow for additional site-specific elements to be added. In order to define the elements for this field, an obligation Business Object may be defined on the associated [obligation type](#).

**Base business object.** The base product provides a business object for a filing period based obligations. Refer to the business object metadata for more information.

## Premise Renamed Location

In this release the term "premise" has been renamed to "location".

## Location Enhancements

In this release, [location](#) has been enhanced to include a CLOB field to allow for additional site-specific elements to be added. In order to define the elements for this field, a location Business Object may be defined on the associated [location type](#).

## Payment Enhancements

---

### Store Characteristics on Payment Tender and Payment Objects

The Tenders page provides a section for adding characteristics to the Payment Tender object, and the Payment page includes a Characteristics tab for adding and deleting characteristics to a Payment object. You can use these characteristics to store implementation-specific information and/or for special processing.

## Financial Enhancements

---

### Contents

- Penalty & Interest Introduced
- Dynamic Credit Allocation Introduced
- Increased GL Account Field Length
- Hold Option Added to A/P Requests
- Support for Direct Deposit Transactions

### Penalty & Interest Introduced

In this release functionality is provided to calculate penalty and interest rules.

For more information, refer to [The Big Picture of Penalty and Interest](#).

### Dynamic Credit Allocation Introduced

Credit allocation is the process of taking credits within an obligation and applying business rules to allocate them against tax, penalty, interest, and fees (in other words, allocated by debt category).

We use the term 'dynamic credit allocation' to indicate that credit allocation should be re-evaluated any time penalty and interest is calculated, whether for updates or forecasting.

For more information, refer to [The Big Picture of Credit Allocation](#).

### Increased GL Account Field Length

Increases GL Account field length to allow for more Chart Fields to be configured with a delimiter.

### Hold Option Added to A/P Requests

The [A/P Request](#) record used to interface accounts payable requests to your external financial system has been enhanced to support a value of **Hold** in the **Payment Selection Status**. This allows your external system to indicate when the check request is on hold in that system.



## Support for Direct Deposit Transactions

In this release, the process that produces the flat file for the Automated Clearing House interface has been updated to support direct deposits. The change was to the population of the TRANSACTION-CD on record type **6** (Entry Detail Record) to cater for transaction codes for checking, savings and credit card deposits.

Refer to [ACH Record Layouts](#) for more information.

## Billing Enhancements

---

### Provide Business Services to Call Common Billing Routines

The following business services have been supplied with the product to enable common billing functions to be performed by service scripts:

- *C1-CancelRebillFreezeBillSegs*
- *C1-CompleteBill*
- *C1-CreateCreditNote*
- *C1-GenerateFreezeBillSegments*

## Adjustment Enhancements

---

### Define Adjustment Approvals

Adjustment approval logic has been provided using the new [approval](#) capability.

## Collections

---

### Contents

- Overdue Monitor
- Overdue Process
- Compliance Rating
- Cash Only Functionality Depreciated
- Support for Collection Cases

### Overdue Monitor

This release provides new base package algorithms to support account compliance review when a credit FT is frozen and an overdue monitor rule that determines if an account has overdue assessments.

For more information, refer to <a href="#">How Does The Overdue Monitor Work</a> .
--

## Overdue Process

In this release, the "base date" field on overdue process has been deprecated.

This release provides new base package algorithms to support overdue processes that collect on overdue assessments, as follows:

- An **Overdue Process Template - Cancel Criteria** algorithm that calculates the unpaid balance on assessments linked to an overdue process and compares to a threshold amount.
- An **Overdue Process Template - Cancel Logic** algorithm that closes open collection cases linked to the overdue process.
- New **Overdue Event Type - Event Activation** algorithms to:
  - Create a collection case for the overdue process's account's main person.
  - Bring P&I up to date for the obligations linked to an overdue process's assessments
  - Write off the unpaid balance of the overdue process's assessments, subject to a threshold limit
- An **Overdue Event Type - Monitor Waiting Events** algorithm that determines if all collection cases linked to an overdue process are complete.

For more information, refer to [The Big Picture Of Overdue Processes](#) and [The Big Picture Of Overdue Events](#).

For information on collection cases, refer to [Support for Collection Cases](#).

## Compliance Rating

In this release, the terms "credit rating" and "credit review" were renamed "compliance rating" and "compliance review", respectively.

## Cash Only Functionality Deprecated

In this release, the "cash-only" score functionality has been deprecated.

## Support for Collection Cases

In this release, functionality is provided to support collection cases. A collection case provides the functionality to record and track the interactions between the taxpayer and the user responsible for the collection of unpaid debt.

For more information, refer to [The Big Picture of Collection Cases](#).

## Pay Plan Enhancements

---

In this release, a number of enhancements have been provided to further support pay plans functionality:

### Contents

- User Interface
- Covered Obligations
- Scheduled Payments Status
- Algorithms
- Collections Information Zone

## User Interface

The following enhancements have been made to the pay plan user interface:

- A Cancel button is introduced on the pay plan page to support miscellaneous reasons for canceling a pay plan. The Cancel button will cause the pay plan to be updated to Canceled status.
- A new tab has been added to the pay plan page to display the pay plan's characteristic collection.
- The main taxpayer ID is now displayed on the pay plan page.
- The search page for selecting covered obligations was changed to allow broader search of the taxpayer's obligation across different accounts.
- The scheduled payment's status is displayed in the scheduled payments grid.

## Covered Obligations

The restriction for covered obligations of a pay plan having to belong to the same account has been removed. A pay plan can cover obligations across different accounts as long as the main taxpayer for these accounts is the same. The main taxpayer on the pay plan's account is now used as the restricting criteria for the search instead of the pay plan's account.

In addition, covered obligations will no longer be unlinked when the pay plan is stopped.

## Scheduled Payments Status

A status was added to the pay plan's scheduled payments. The scheduled payments will be created in a **pending** status and will be updated by the base package batch to **processed** status after processing the scheduled payments on the payment due date. When processing the scheduled payments, the batch will take into consideration the credit review grace days from the account type.

In addition, scheduled payments will no longer be deleted after the scheduled payment is processed.

## Algorithms

A number of new base package algorithm types have been provided in this release:

- Two **Pay Plan Recommendation Schedule - Generate Payment Schedule** algorithms that generate a payment schedule are provided: fixed amount payment schedule or a fixed duration payment schedule.

- An ***Obligation Type - Process Pay Plan Scheduled Payment*** algorithm that will create a to do if a scheduled payment has not been met, or if the pay plan has been paid off.
- An ***Obligation Type - Obligation Cancel*** algorithm that will create a customer contact when a pay plan is cancelled.
- An ***Obligation Type - Break Pay Plan*** algorithm that will create a to do if the pay plan is manually broken and the pay plan is not associated with a collections case.
- An ***Obligation Type - Obligation Activation*** algorithm that will create a setup fee when the pay plan is activated.
- A ***Distribution Rule - Create Payment*** algorithm that will create payments for obligations covered by the pay plan based on age of debt and obligation priority.

## Collections Information Zone

The Collections Information Zone has been enhanced to include one row for each active pay plan that is linked to the account.

For more information, refer to [The Big Picture of Pay Plans](#).

## Support for Overpayment Processing

In this release, functionality is provided to support overpayment processing. An overpayment process provides the functionality to manage the approval, and allocation of the credit amount based on the tax agency's rules.

For more information, refer to [The Big Picture of Overpayments](#).

## Forms Processing

Forms processing functionality is introduced in this release. This provides ability to store and process registration forms, tax forms and form batches (for manual data entry).

For more information, refer to [The Big Picture of Registration](#), [The Big Picture of Tax Forms](#) and [The Big Picture of Form Batch Processing](#).

## Miscellaneous Enhancements

### Contents

[Sample and Submit Approvals](#)

## Masking Sensitive Data Terminology Changes

### Sample and Submit

The [sample and submit](#) feature enables an implementer to design a process to select records based on appropriate criteria and then perform actions for the business process, for example " send a letter to all taxpayers eligible for a stimulus tax rebate".

### Approvals

You now have the ability to incorporate approval and sign off actions into your business processes. In addition, logic for approving adjustments has been provided.

For more information, refer to [The Big Picture of Approvals](#).

### Masking Sensitive Data

As described in Data Masking the ability to mask sensitive and confidential data is supported. In Oracle Enterprise Taxation Management, this functionality can be used to [mask data](#) such as personal identification numbers, credit cards and bank account numbers.

### Terminology Changes

The following terms were changed for this release:

- **A/P 1099 Flag** was changed to **Income Statement**.
- **Arrears Date** was changed to **Effective Date**.
- **Bill Factor (BF)** was changed to **Rate Factor (RF)**.
- **Service Quantity (SQ)** was changed to **Rate Quantity (RQ)**.
- **Service Quantity Indicator (SQI)** was changed to **Rate Quantity Indicator (RQI)**.

Other term changes are mentioned in appropriate sections above.

### Asset Functionality Deprecation

The asset functionality released previously is no longer supported. This includes the following:

- Asset Type
- Asset
- Manufacturer
- Asset rate component type functionality

## New Application Services

The following table contains the new application services made available in this release. You must link these services to your [user groups](#) if you want the users in these groups to have access to these pages / zones.

Application Service ID	Description of Service	Permissible Actions
<i>C1ACMACR</i>	Sample & Submit Request Portal	Inquire
<i>C1ACMACT</i>	Activity Type Portal	Inquire
<i>C1ACMARQ</i>	Sample & Submit Request Query Portal	Inquire
<i>C1-ADJAPPRVLREQBOAS</i>	Adjustment Approval Request Business Object	Add, Change, Approved, Determine Approvers, Approval In Progress, No Approval Necessary, Rejected, Delete, Inquire
<i>C1-APAYEXPCRREQBOAS</i>	Activity Request - Expire AutoPay Credit Card Business Object	Add, Change, Pending, In Progress, Cancel, Complete, Delete, Inquire
<i>C1-APAYEXPCRTYPEBOAS</i>	Activity Type - Expiring AutoPay Credit Card Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1APROF</i>	Approval Profile Portal	Inquire
<i>C1ARTYPE</i>	Asset Relationship Type Portal	Inquire
<i>C1ASSETP</i>	Asset Portal	Inquire
<i>C1ASSETQ</i>	Asset Query Portal	Inquire
<i>C1ASTYPE</i>	Asset Type Portal	Inquire
<i>C1-BUSREGFORMBOAS</i>	Business Registration Form Business Object	Add, Change, Canceled, Pending, Posted, Review, Suspended, Validate, Waiting for Information, Delete, Inquire
<i>C1-BUSREGFRMTYBO</i>	Business Registration Form Type Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1CLCQRY</i>	Collection Case Query Portal	Inquire
<i>C1CLCSTY</i>	Collection Case Type Portal	Inquire
<i>C1COLCAS</i>	Collection Case Portal	Inquire
<i>C1DBCTPR</i>	Debt Category Priority Portal	Inquire
<i>C1DBTCAT</i>	Debt Category Portal	Inquire
<i>C1FBH</i>	Form Batch Header Portal	Inquire
<i>C1FBHQ</i>	Form Batch Header Query Portal	Inquire
<i>C1-FORMBATCHHEADER</i>	Form Batch Header Maintenance Object	Add, Change, Delete, Inquire
<i>C1-FORMTYPEM</i>	Form Type Maintenance Object	Add, Change, Delete, Inquire
<i>C1FRTM</i>	Form Type Portal	Add, Change, Delete, Inquire

<b>Application Service ID</b>	<b>Description of Service</b>	<b>Permissible Actions</b>
<i>C1FRTYSP</i>	Form Type Query Portal	Inquire
<i>C1-GENFORMBATCHHDR</i>	Form Batch Header Business Object	Add, Change, Pending, Validated, Complete, Delete, Inquire
<i>C1-INDFRMTYPEBOAS</i>	Individual Income Tax Form Type Business Object	Add, Activate, Change, Delete, Inactivate, Inquire
<i>C1-INDXFRMBOAS</i>	Individual Income EZ Tax Form Business Object	Add, Change, Adjusted, Canceled, Pending Adjust, Pending, Posted, Pending Transfer, Pending Reversal, Reversed, Suspended, Transferred, Validate, Waiting for Information, Delete, Inquire
<i>C1LABALP</i>	Control Central - Credit Allocation Detail (Acct Info) Zone	Inquire
<i>C1LTBALP</i>	Control Central - Credit Allocation Detail (Taxpayer) Zone	Inquire
<i>C1-ONWAIVERTYBO</i>	Ongoing Waiver Type Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1OPTYPE</i>	Overpayment Process Type Portal	Inquire
<i>C1-OTWAIVERTYBO</i>	One Time Waiver Type Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1OVPP</i>	Overpayment Process Portal	Inquire
<i>C1OVPPQ</i>	Overpayment Process Query Portal	Inquire
<i>C1-OVPPROCBOAS</i>	Individual Taxpayer Overpayment Process Business Object	Add, Change, Approved, Canceled, Decision, Approval In Progress, Minimum Amount Write Off, Rejected, Validate, Complete, Delete, Issue, Inquire
<i>C1-OVPPROCM</i>	Overpayment Process Maintenance Object	Add, Change, Delete, Inquire
<i>C1-OVPPRTYBOAS</i>	Individual Taxpayer Overpayment Process Type Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1-OVRPAYTYPM</i>	Overpayment Process Type Maintenance Object	Add, Change, Delete, Inquire
<i>C1PERTYP</i>	Person Type Portal	Inquire
<i>C1PICTRL</i>	P&I Control Portal	Inquire
<i>C1PIRULE</i>	P&I Rule Portal	Inquire
<i>C1RGFRM</i>	Registration Form Portal	Inquire
<i>C1RGFRMQ</i>	Registration Form Query Portal	Inquire
<i>C1-PICONTROL</i>	P&I Control Maintenance Object	Add, Change, Delete, Inquire

Application Service ID	Description of Service	Permissible Actions
<i>C1-PIRULE</i>	P&I Rule Maintenance Object	Add, Change, Delete, Inquire
<i>C1-STANDARDPICONTROL</i>	Standard P&I Control Business Object	Add, Change, Active, Inactive, Pending, Delete, Inquire
<i>C1-STNDCOLLCASEBO</i>	Standard Collection Case Business Object	Add, Change, Actions In Progress, Closed, Pending Investigation, Transferred, Uncollectible, Delete, Inquire
<i>C1-STNDCOLLCASETYBO</i>	Standard Collection Case Type Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1-SUTXFRMBOAS</i>	Sales and Use Short Tax Form Business Object	Add, Change, Adjusted, Canceled, Pending Adjust, Pending, Posted, Pending Transfer, Pending Reversal, Reversed, Suspended, Transferred, Validate, Waiting for Information, Delete, Inquire
<i>C1-SUTXFRMTYBOAS</i>	Sales and Use Short Tax Form Type Business Object	Add, Change, Active, Inactive, Delete, Inquire
<i>C1-TAXFORMBOAS</i>	Parent Tax Form Business Object	Add, Change, Adjusted, Canceled, Pending Adjust, Pending, Posted, Pending Transfer, Pending Reversal, Reversed, Suspended, Transferred, Validate, Waiting for Information, Delete, Inquire
<i>C1-TAXFORMMM</i>	Tax Form Maintenance Object	Add, Change, Inquire
<i>C1TXFRM</i>	Tax Form Portal	Inquire
<i>C1TXFRMQ</i>	Tax Form Query Portal	Inquire
<i>C1TXROLE</i>	Tax Role Portal	Inquire
<i>C1TXRQRY</i>	Tax Role Query Portal	Inquire
<i>C1TXTYPE</i>	Tax Type Portal	Inquire
<i>C1-WAIVERBO</i>	Waiver Business Object	Add, Change, Active, Pending, Cancel, Delete, Inquire
<i>C1-WAIVERM</i>	Waiver Maintenance Object	Add, Change, Delete, Inquire
<i>C1-WAIVERTYM</i>	Waiver Type Maintenance Object	Add, Change, Delete, Inquire
<i>C1WAIVP</i>	Waiver Portal	Inquire
<i>C1WAIVQ</i>	Waiver Query Portal	Inquire
<i>C1WAIVTY</i>	Waiver Type Portal	Inquire
<i>CILCAMRP</i>	Activity Request Maintenance Object	Add, Change, Delete, Execute, Inquire
<i>CILCPMPP</i>	Location Management	Add, Change, Assign Parent



Application Service ID	Description of Service	Permissible Actions
		Location, Remove Parent Location, Inquire
<i>CILCRGFP</i>	Registration Form Maintenance Object	Add, Change, Delete, Inquire
<i>CILCTAXP</i>	Tax Role Maintenance Object	Add, Change, Delete, Inquire
<i>CILLCCSP</i>	Collection Case Maintenance Object	Add, Change, Delete, Inquire
<i>CILMITMP</i>	Asset	Add, Change, Delete, Inquire
<i>CILMITMP</i>	Item	Add, Change, Delete, Inquire
<i>CILMONPP</i>	Monitor Pay Plan Service	Add, Change, Delete, Inquire
<i>CILPCAPP</i>	Create One-Off Automatic Payment	Add, Change,
<i>CILTAMTP</i>	Activity Type Maintenance Object	Add, Change, Delete, Execute, Inquire
<i>CILTAPFP</i>	Approval Profile Maintenance Object	Add, Change, Delete, Inquire
<i>CILTCCYP</i>	Collection Case Type Maintenance Object	Add, Change, Delete, Inquire
<i>CILTDBCP</i>	Debt Category Maintenance Object	Add, Change, Delete, Inquire
<i>CILTDCPP</i>	Debt Category Priority Maintenance Object	Add, Change, Delete, Inquire
<i>CILTFCAP</i>	Filing Calendar	Add, Change, Delete, Inquire
<i>CILTITTP</i>	Asset Type	Add, Change, Delete, Inquire
<i>CILTITTP</i>	Item Type	Add, Change, Delete, Inquire
<i>CILTPTYP</i>	Person Type Maintenance Object	Add, Change, Delete, Inquire
<i>CILTXMLP</i>	Add XML Extract Service	Add

## Changed Application Services

The action **Submit for Approval** has been added to the application service *CILAADUP* (Adjustment).

The action **Cancel** has been added to the application service *CILCNBBP* (Pay Plan).

## Application Services No Longer Applicable

The application services *CILMITMP* and *CILTITTP* are no longer applicable. These were related to the Asset functionality previously released that is now deprecated.

## User Exit Deprecation

All "Read" action user exit points in COBOL Row Maintenance are no longer supported.

## **Planned Background Process Deprecation**

---

The SAACT (Obligation Activation) background process will be deprecated in a future release. The functionality provided in this background process is not applicable.