Anthology

Anthology Payroll for Microsoft Dynamics 365 Finance

Anthology Payroll Import Application Program Interface Guide

Date: March 2025

www.anthology.com

Contents

Introduction 4
Integration between Project and Anthology Payroll 4
General Processing Flow 5
Considerations 7
Timing with Pay Processing in Anthology Payroll 7
Other Considerations 8
Development and Code Verification 9
Multiple Dynamic API Instances 9
View Data in Dynamics 10
Import Trial Data 10
API Specifications 11
PHRImport::construct 11
PHRImport::getPositions 12
PHRImport::getPayGroup 12
PHRImport.init 12
PHRImport.importValidate 14
PHRImport.getNumOfExceptions 14
PHRImport.getExceptions 15
PHRImport.importCommit 15
PHRImport.importFinalize 16
PHRImport.setOrgSplitOverride 16
Parameters for Import Methods 17
Worker Identification 17
Position Identification 18
Date 18
Start Time 19
End Time 20
Hours 20
Units 21
Shift 22
Earnings 22
Piece 23
Amount 23
Salary 24
Ledger 24

Department 25
Dimension 25
Occupation 26
Job 26
Reasons 28
Benefit/deduction Amount 28
Coding Samples 30
Add Time or Earnings 30
Frequently Asked Questions 32
Affected Tables 34

Introduction

This document describes the Anthology Payroll Import application programming interface (API) for Microsoft Dynamics 365 Finance. It is an X++ interface. You can use it to extend Microsoft Dynamics 365 Finance modules, usually Project, to write time, earnings, and to import scheduled benefit/deduction entries into Anthology Payroll tables.

A typical customization using this API would:

- Provide a user interface for recording time, earnings, scheduled benefit/deductions or all of the above
- Validate those entries, present error messages and ensure the entries are valid before saving them
- Provide a user interface for posting those entries to a module like Project, as well as Anthology Payroll. At that time, your custom code would validate them again, just in case anything has changed, and either transmit them to Anthology Payroll or provide feedback about errors.

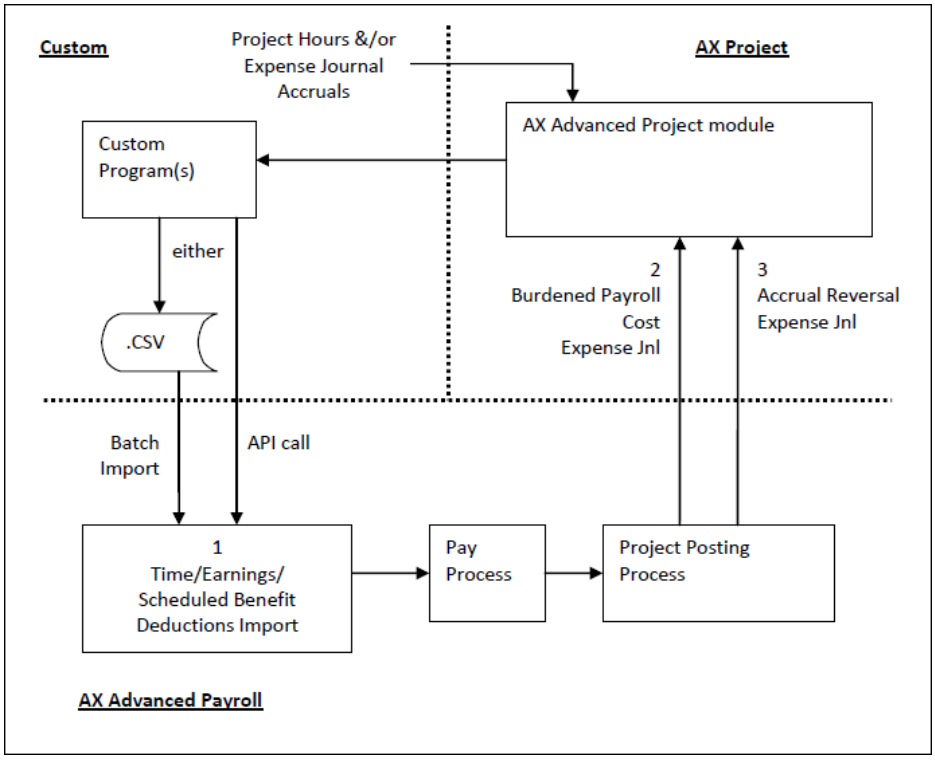
This document will guide you in:

- Transmitting time, earnings and scheduled benefit/deduction records to Anthology Payroll
- Managing data validation
- Verifying that your code is working

Integration between Project and Anthology Payroll

The Anthology Payroll Import API complements the Anthology Payroll integration to Project. This API provides a mechanism for transmitting time and earnings data to Anthology Payroll. The Anthology Payroll integration to Project, in turn, updates Project expense journals with actual expense amounts from payroll:

- The Project posting functionality in Anthology Payroll transmits actual earning plus employer benefits costs to a Project expense journal. These actual amounts can either update the original journal entries that were transmitted to Anthology Payroll or appear as new Project journal entries. A Payroll parameter determines whether new entries represent a maximum of one day or one pay period.
- If the Project posting functionality in Anthology Payroll creates new journal entries in Project, it can also create reversal entries for the time and earnings journal entries that were originally transmitted to Anthology Payroll. This means that you can accrue estimated expenses as they are entered in Project, then reverse them when the actual expenses, including benefits costs, are posted from Anthology Payroll at the end of the pay period.



***Note**: The **Project** and **Category** fields from the Project module are required for the Anthology Payroll integration to Project. The **Journal** and **Line** fields are required if you configure Anthology Payroll to update the original Project journal entries with actual payroll costs. The **Journal**, **Line** and **Reversal Category** are required if you configure Anthology Payroll to reverse the original Project journal entries when adding actual payroll costs as new journal entries.*

General Processing Flow

You can import time records, earning records, scheduled benefit/deduction records, or all into Anthology Payroll via the Anthology Payroll Import API. Transactions imported via this API must include:

- Pay group. All processing in the Anthology Payroll module is done within the context of a pay group. A pay group has a currency. All earnings transactions in a pay group share its currency. A pay group also has a current pay period. All transactions must be dated on or before the end date of that current pay period.
- The worker for whom time or earnings information is being sent.
- If a worker has more than one position, time and earnings will be recorded against the worker's primary position, unless you specify otherwise.
- The date of the time or earnings record.
- Time records can be expressed as one of:
  - hours
  - start and end time
  - units produced and piece information
- Earnings records can be expressed as one of:

    - Amount earned
    - Amount rate and hours
    - Amount rate and units
    - Salary grid and step

Scheduled benefit/deduction records imported via this API must include:

- Pay group. All processing in the Anthology Payroll module is done within the context of a pay group. A pay group has a currency. All earnings transactions in a pay group share its currency. A pay group also has a current pay period. All transactions must be dated on or before the end date of that current pay period.
- The worker for whom scheduled benefit/deduction information is being sent.
- Pay period end date in which the benefit/deduction is to be processed in.
- Benefit/deduction records can be expressed as one of:
    - Worker amount and/or employer amount
    - Worker calculation and/or employer calculation

In one call, you must pass all time for all workers in a pay group before you pass time entries for another pay group. You must pass all time for all workers before you pass earnings. You must pass all earnings for all workers in a pay group before you pass entries for another pay group. You cannot mix time and earnings for the same worker, nor can you mix pay groups.

The general process flow is as follows:

1. Call construct to specify the type of import information (time or earnings).
2. Call **init** to change the pay group.
3. To derive a worker's pay group, call **getPayGroup**.
4. To derive a worker's position, call **getPositions**.
5. Call **importValidate** (or **importCommit**, depending on requirement) to validate the time, earning, or benefit/deduction information provided and import the records to Anthology Payroll tables. If it finds any exceptions, then it sets the return status to False. If it finds no exceptions, then it sets the return status to True and inserts the transactions into the relevant tables.
6. If **importValidate** (or **importCommit**) finds exceptions, it does not insert transactions into the Anthology Payroll import tables. Call **getNumOfExceptions** to determine the number of exceptions, then call **getExceptions** to retrieve those exception message(s). You will need to develop some kind of user interface to communicate these exceptions to the end user. The exceptions indicate what must be resolved before the transactions can be successfully imported to Anthology Payroll tables.
7. Depending on your implementation's process flow, one of the following:
    a. Your payroll administration reviews the imported transactions from within Microsoft Dynamics 365 Finance and proceeds with processing them as part of the normal pay process.
    b. Call **importFinalize** to log off the Anthology Payroll assemblies.

Time records are imported into the **PHRWorkerTimeImports** and **PHRWorkerTime** tables. Earning records are imported into the **PHRWorkerEarnImports** and **PHRWorkerEarns** tables.

Benefit/deduction records are imported into the **PHRWorkerBenDedImports** and **PHRWorkerBenDeds** tables.

Time, earnings, and benefit/deduction transactions imported via the API are committed and approved when the **importCommit** method is called. Records that fail validation are not committed into Anthology Payroll. The **importValidate** method will import and validate all transactions but leave them in an uncommitted state where they are not yet live transactions that can be used in pay processing. These transactions can be committed from the Dynamics client in the respective import journal depending on the type of transaction (**PHRCommitTime**, **PHRCommitEarn**, and **PHRCommitBenDedAmts**).

Considerations

The following considerations should be considered when developing a solution that uses the Anthology Payroll API.

Timing with Pay Processing in Anthology Payroll

The Anthology Payroll import API allows you to import transactions and commit them for pay processing, but this may not be the best practice for your organization. If you are developing a solution to import transactions from another Dynamics module, those who are responsible for the import may be unaware of the current status of the pay process when importing transactions. As such, it is generally recommended not to call the **importCommit** and use **importValidate** to import your transactions instead. This allows your payroll administrators to manage these transactions from the Dynamics client before they become live in the system.

The **importValidate** method imports transactions to Anthology Payroll import tables regardless of whether they are valid or not (returns True or False). Invalid transactions can be managed in the following ways:

- call the **finalize** method to delete them from the import tables before committing them. This deletes all transactions from the import tables. You may call the **getNumOfExceptions** and **getExceptions** methods to determine the errors and resolve them before importing again.
- allow payroll administration to modify or delete these transactions from the corresponding journals in the Dynamics client.

Once processing of a payroll period has begun in Anthology Payroll, you need to be careful about importing additional records into Anthology Payroll. Imported transactions can cause problems for pay calculation if they are imported at the wrong time. For example, if payments have already been calculated and issued for the pay period and payroll is ready to close the pay period, committed transactions will prevent the pay period from being closed until the transactions are dealt with. It is generally recommended that the process of verifying and committing transactions to Anthology Payroll processing tables be left for payroll administration to perform.

Anthology Payroll can also be set up to lock out transaction record imports after time records, earnings, and benefit/deductions have been calculated. Use the Pay processing locking parameter on the **General definitions** form (**Setup** > **Parameters**). If Pay processing locking is enabled in Anthology Payroll, transaction records cannot be imported if pay processing for those records have already begun for the current pay period. In this case, transactions can still be entered manually to the current pay period as adjustments. You may consider creating a report for your payroll administration to aid with manual data entry in Anthology Payroll and a mechanism for marking these records as not needing to be included in future imports into Anthology Payroll.

Other Considerations

The following table describes additional considerations:

Time or earnings	Typically, you'll import time records into Anthology Payroll and let Anthology Payroll use current wages and salaries to calculate the appropriate earnings. You'll usually only import earnings where the amount is ad hoc, as in the case of reimbursable expenses. If you are writing your code as part of an integration between Project and Anthology Payroll, for salaried workers, transmit the number of hours in a full work day for each day worked (time entries are limited to a single day). A day would be broken up into multiple time records if the worker worked on multiple projects or categories during the day.
Scheduled benefit/ deductions	When you schedule a benefit/deduction, it will override any regularly scheduled benefit/deduction and its calculation for that pay period if it already exists as an worker benefit/deduction. It is not an amount in addition to the amount that would be regularly generated.
Date	Ensure that all time and earnings that you transmit to Anthology Payroll are dated on or before the last date of the current pay period for the pay group you are transmitting. If you transmit an entry dated after the current pay period, it will not be processed in either the current pay period or the next pay period. You can submit new records for past pay periods. You might want to do this if data missed getting entered in a past pay period and it didn't get automatically generated by the Anthology Payroll module. Pay group information is stored in the **PHRPayGrps** table. Code samples at the end of this document demonstrate how to look up a pay period's end date. Date format must be as specified in this document.
Currency	If you have transactions in multiple currencies, you should verify that any earnings amounts you send to Anthology Payroll are transmitted in the currency that Anthology Payroll requires. When you transmit earnings, you associate them with a pay group, and each pay group has a currency. You should verify the pay group's currency and ensure that the earnings you transmit are in that currency.
Workers	Microsoft Dynamics 365 Finance maintains a central list of all workers in all companies.

	Anthology Payroll recognizes only the workers that have been configured for Anthology Payroll in the company that you are currently importing into. It is possible that a worker could be set up for Project, but not Anthology Payroll. If you encounter this situation, your code should identify the exception to the user.
Record which data has been transmitted to Anthology Payroll	As each record is successfully imported data into Anthology Payroll, lock or otherwise mark it so that you don't import it again and create duplicate records in Anthology Payroll. Records that fail validation and are not imported should be manually corrected and re-submitted to Anthology Payroll via the API.

Development and Code Verification

This section contains guidelines and tips on developing your solution and verifying your code.

Multiple Dynamic API Instances

Each instance of the API will contain either time, earning, or benefit/deduction records for one pay group. If you want to ensure that all your transactions or your pay group records are valid before you commit any of them, you need multiple API instances running in parallel.

For each time, earning, and benefit/deduction record that you want to import into Anthology Payroll:

1. Determine the record type: time, earning, or benefit/deduction.
2. Determine the pay group by looking up the pay group of the position or, if there is no position specified, the worker associated with that record.
3. Determine whether you already have an API instance for that pay group and record type. If you don't:
    a. Create a new API instance for that pay group and record type.
    b. If this is the first API instance for the pay group:
        - Look up the end date of the current pay period for the pay group.
        - If the API instance is for earnings or deductions, look up the currency of the pay group.
4. To process a time record for a particular pay group:
    a. Verify that the date of the time record is on or before the end date of the current pay period for the pay group.
    b. If verification was successful, invoke **validateAndInsert** using the API instance appropriate for the record's pay group and record type.
5. To process an earning record for a particular pay group, its API instance must:
    a. Verify that the date of the earnings record is on or before the end date of the current pay period for the pay group.
    b. Verify that the currency of the earnings record matches the currency of the pay period. If it doesn't, either generate an error message or convert the earnings into the pay group's currency.
    c. If verification was successful, invoke **validateAndInsert** using the API instance appropriate for the record's pay group and record type.
6. If you encounter a problem and need to roll back, close all the API instances by calling **finalize** without calling **commit** in all API instances.

7. When you have processed all records and want to commit them to Anthology Payroll as transactions to process, call **commit** and then **finalize** in all API instances. Alternatively, you may forgo calling these methods and allow payroll administration to manually modify or commit the transactions from the Dynamics client.

View Data in Dynamics

When you start transmitting data to Anthology Payroll via the API, you should review it in the Dynamics client to confirm it arrived as you expected.

To see time entries, click **Anthology Payroll** > **Inquiries** > **Time import** in the navigation pane to open the Time imports inquiry.

To see earning entries, click **Anthology Payroll** > **Inquiries** > **Earnings import** in the navigation pane to open the **Earnings imports inquiry**.

To see scheduled benefit/deduction entries, click **Anthology Payroll** > **Inquiries** > **Benefit/deduction amounts import** in the navigation pane to open the **Benefit/deduction amounts imports inquiry**.

Import Trial Data

If you are getting data validation errors and want to confirm whether the problem really is with the data or with your use of the API, try importing the data from a comma-delimited file.

The comma-delimited file should have a header row that uses the column headings as listed in **Parameters for Import Methods** on page 13 to identify which columns are being imported and in which order. All text fields must be surrounded by double quotes. Numeric and date fields do not require double quotes.

1. Import your data through one of the following:
    a. time: **Import time** form accessed from **Periodic** > **Pay Period** > **Import time**
    b. earnings: **Import earnings** form accessed from **Periodic** > **Pay Period** > **Import earnings**
    c. benefit/deductions: **Import benefit/deduction amounts** form accessed from **Periodic** > **Pay Period** > **Import benefit/deduction amounts**
2. View the imported data through one of the following:
    a. time: **Time import inquiry** accessed from **Inquiries** > **Time import**
    b. earnings: **Earnings import inquiry** accessed from **Inquiries** > **Earnings import**
    c. benefit/deductions: **Benefit/deduction amounts import inquiry** accessed from **Inquiries** > **Benefit/deduction amounts import**

If the data imports cleanly, the problem is in your use of the API.

If the import also fails validation, then the problem is with the data. The most elusive data problems relate to Anthology Payroll effective dates. Most objects in Anthology Payroll have start dates and end dates. Each worker has a start and end date. Each worker position has a start and end date. Each shift code and earnings code has a start and end date. So even if a lookup value exists, you could get a validation error because the date for the time entry you are importing is before the lookup value's start date or after its end date.

API Specifications

The Anthology Payroll API is named **PHRImport** and it is a Base class type.

The following table describes the methods included in the **PHRImport** class in the order they are typically called:

Method	Description
construct	Constructs the import object and specifies whether time or earnings are being imported.
getPositions	Returns a worker's positions
getPayGroup	Returns a worker's pay group
init	Specifies the pay group being imported and which fields are being imported
importValidate	Validates the data being imported and, if valid, inserts it into Anthology Payroll staging tables where they can be reviewed in the Dynamics client application
getNumOfExceptions	Returns the number of exceptions identified in a **validateAndInsert** call
getExceptions	Returns the exception messages generated in a **validateAndInsert** call
importCommit	Imports, validates and commits transactions to Anthology Payroll
importFinalize	Imports, validates and commits transactions to Anthology Payroll completes the API call and cleans up any staging tables.
setOrgSplitOverride	Overrides Anthology Payroll's default setting for whether Department, Ledger, Dimension or Project/Category/Activity values being imported override competing default values configured in Organization splits.

The following sections describe the API method parameters, method types, and return values, and how to use each method.

PHRImport::construct

Method Parameters	PHRImportType
Method Type	Static
Return Value	PHRImport

The construct method must pass in the *PHRImportType* parameter. The *PHRImportType* is an enumerator, and the valid enumerations are:

- `None`
- `Time`
- `Earnings`
- `BenDed`

To send time information, pass Time; to send earning information, pass Earnings; to send benefit/deduction information, pass BenDed.

If the parameter contains any other value, the method returns the following message:

```
The import type is invalid. The valid types are Earnings, Time, or
Benefit/deduction.
```

Processing ceases and no object is constructed.

PHRImport::getPositions

Method Parameters	HcmWorkerRecId, PHRActiveDate
Method Type	Static
Return Value	Container (multiple positions and their attributes)

The method returns all the positions held by the worker you identify in the input parameters. If you do not provide a *PHRActiveDate*, then the current system date is used.

You must provide the *HcmWorkerRecId*. If you do not provide it, then an empty container is returned.

The returned container contains the following fields in the following order: position rec id, pay group code, position description, job rec id, department rec id, occupation code, union affiliation code, union certification code, active date and expiry date.

PHRImport::getPayGroup

Method Parameters	HcmWorkerRecId, HcmPositionRecId, PHRActiveDate
Method Type	Static
Return Value	PHRPayGrpCode

This method returns the pay group for the worker you specify in the input parameters. If you do not provide a *PHRActiveDate*, then the current system date is used.

If you do not provide any parameters and only one pay group code is setup in Anthology Payroll, then the method returns that pay group code. If multiple pay groups are set up, then no pay group is returned unless you provide parameters.

If you provide an *HcmPositionRecId*, but that position cannot be found for the worker, or you do not provide an *HcmPositionRecId* at all, then this method returns the pay group for the worker's primary position.

If no pay group can be determined, then an empty string is returned.

PHRImport.init

Method Parameters	PHRPayGrpCode, PHRImportFieldStr
Method Type	Instance
Return Value	*True* or *False*

This method creates run control information in the Anthology Payroll module for tracking information imported via method calls. It also determines in which pay period the imported time, earnings or benefit/deductions are processed. Finally, it establishes the data elements and the order in which you provide them to the **validateAndInsert** method

You must provide a pay group code. Each pay group has a current pay period into which your transactions are imported. That current pay period must be open in Anthology Payroll to accept your imported data. If you receive a False return value, the error message specifies the problem.

Error message	Resolution
The pay group must be provided.	Provide a pay group in your **init** call.
The pay processing has already been completed for the pay group's current pay period. No time or earnings can be imported.	Pay processing has been completed for the pay group's current pay period. The information must be manually entered in Anthology Payroll to be included in the current pay period. If you enter them manually, you will need a mechanism to ensure those records aren't submitted for processing in the next pay period.
The pay run is currently in progress. No time entry [or earnings entry] is allowed.	Pay processing is in progress, which locks time or earnings entry. You might be able to import your data when the process is finished, unless Anthology Payroll is set up to use payment process locking. In that case, the information must be manually entered in Anthology Payroll. If you enter them manually, you will need a mechanism to ensure those records aren't submitted for processing in the next pay period.

The *PHRImportFieldStr* parameter can optionally contain a list of the field names and the order in which you will be including them in the container you provide to the **validateAndInsert** method. The valid field names appear in Parameters for Import Methods on page 13. The field names can be in any order, separated by commas. If you do not supply any field names in this parameter, then your **validateAndInsert** container must include all fields in the order specified in Parameters for Import Methods on page 13.

A minimal valid earning transaction can be created with the following fields:

- Worker ID
- Date
- Earning
- Amount

A minimal valid time transaction can be created with the following fields:

- Worker ID
- Date
- Hours

A minimal valid benefit/deduction transaction can be created with the following fields:

- Worker ID
- Benefit/deduction
- One of:
    - Worker amount and/or Employer amount
    - Worker calculation and/or Employer calculation

Field names are case-insensitive. If any of the field names provided are invalid, then an error message is returned and the return status is set to False.

If the pay group is valid, its current pay period can accept time and earnings, and all field names are valid, then the return status is set to True. Otherwise the return status is set to False.

PHRImport.importValidate

Method Parameters	Container
Method Type	Instance
Return Value	*True* or *False*

This method takes all of the parameters supplied in the container and validates them within the context of Anthology Payroll. If the information for a time, earning or benefit/deduction record is valid, then this method returns a True return status and the information is inserted into Anthology Payroll import tables for payroll processing. Transactions are imported to the following tables depending on the type:

- time: *PHRWorkerTimeImports*
- earnings: *PHRWorkerEarnImports*
- benefit/deductions: *PHRWorkerBenDedAmtImports*

You should mark that record as having been successfully imported so that you do not try to import it again.

If the information for a time, earning or benefit/deduction record is invalid, then this method returns a False return status. Invalid records are still imported to the corresponding table. You can call **finalize** to clear the import tables or leave the records there for payroll administration to manage.

To determine the reason for the invalid records, call the **getNumOfExceptions** and **getExceptions** methods to retrieve the exception messages. You can then resolve the exceptions, then resubmit the record(s) with another **importValidate** call.

All imported transactions can be viewed and committed to transaction tables from the appropriate import journals in the Dynamics client application.

PHRImport.getNumOfExceptions

Method Parameters	None
Method Type	Instance
Return Value	the number of exceptions as an integer value

This method returns the number of exceptions found on the time or earning information that was just validated in the **validateAndInsert** method.

You only need to call this method if the **validateAndInsert** method returns a False return status.

PHRImport.getExceptions

Method Parameters	Exception number
Method Type	Instance
Return Value	an exception message

This method returns an exception message found for the time, earning, or benefit/deduction information that was just validated in the **validateAndInsert** method.

If the **validateAndInsert** method returns a False return status, you should call the **getNumOfExceptions** method to determine how many exceptions were found, then call this method that number of times to retrieve all the exception messages. You will need to design a user interface to display these messages to the end user, suggest how to correct the problem, and re-send the corrected time, earning, or benefit/deduction information.

If an exception is noted on time, earning, or benefit/deduction information, then that information is not placed into Anthology Payroll tables. Users can correct the exception information either directly in the Anthology Payroll module or in the source system, to be re-sent to Anthology Payroll.

PHRImport.importCommit

Method Parameters	Container
Method Type	Instance
Return Value	*True* or *False*

This method validates, imports and commits the transactions from the container object, where they become live in the payroll system. Once committed, a transaction cannot be undone. This method does everything that the **importValidate** method does with the addition of committing the transactions to the live transaction tables.

Committed transactions are moved to the following tables depending on the type:

- time: *PHRWorkerTime*
- earnings: *PHRWorkerEarns*
- benefit/deductions: *PHRWorkerBenDedScheds*

A True return code means transaction is successfully committed. A False return code means you called **importCommit** without initializing the API by calling **init** or the API was not initialized properly when you made your call to init. The error is displayed in the Infolog. Call **importFinalize** to gracefully clean up the API.

This method is optional. You may choose to call **importValidate** to import transactions to corresponding import tables and modify and commit from the client application or call **importCommit** to perform all those tasks from the API.

PHRImport.importFinalize

Method Parameters	None
Method Type	Instance
Return Value	*True* or *False*

This method logs off the Anthology Payroll assemblies. Any uncommitted records in the *PHRWorkerTimeImports*, *PHRWorkerBenDedAmtImports*, and *PHRWorkerEarnImports* tables remain and can be viewed, modified, deleted or committed from the appropriate journals in the Dynamics client application.

This method should be called after your **importValidate** or **importCommit** method.

PHRImport.setOrgSplitOverride

Method Parameters	*True* or *False*
Method Type	Instance
Return Value	*None*

This method is only relevant when both of the following conditions apply:

- You are importing Earnings (PHRImport.construct)
- You use PHRimport.importCommit. It is not applicable when you use the PHRimport.importValidate method.

If you do not use PHRimport.setOrgSplitOverride in the above situation, then Anthology Payroll's default setting from its Parameters form will determine whether Department, Ledger, Dimension or Project/Category/Activity values being imported override competing default values configured in Organization splits when creating earning transactions. Usually, this method is not needed.

If you pass a parameter of True, then any Department, Ledger, Dimension or Project/Category/Activity values being imported will override any competing default values configured in Organization splits, regardless of the default configuration in Anthology Payroll.

If you pass a parameter of False, then any Department, Ledger, Dimension or Project/Category/Activity values being imported will be dropped when there are competing default values configured in Organization splits, regardless of the default configuration in Anthology Payroll.

Parameters for Import Methods

This topic lists and describes the parameters that the **importValidate**, and **importCommit** method(s) require. In the **init** method, you determine whether the container you pass to the import method contains these parameters in the column order specified below, or whether it contains them in the order specified in your **init** call. Your **init** call should use the exact field names from these tables.

All parameters have the String data type. Unless otherwise specified below, parameters can contain empty strings where you do not want to pass a specific parameter value.

If a parameter you pass to the import method does not meet the validation rules, an exception is returned and the transaction is not imported to the Anthology Payroll database.

These parameters are listed in the order in which they should be passed to the import method.

Worker Identification

There must be at least one piece of information that identifies the worker associated with the imported record. If there is none, an exception is logged. The worker record must be active on the date that you are importing transactions for.

A worker's dates can be viewed on their **Workers** form (**Anthology Payroll** > **Common** > **Anthology Payroll workers** > **select worker** > **Payroll Profile** tab > **Payroll worker**).

Worker IDs (Personnel numbers) are stored in the *HcmWorker* table. Worker names are stored in the

DirPersonName table.

Field	Description	Validation
Worker ID	Worker’s Identification	Alphanumeric, max 20 characters. Must match a corresponding record in Advanced Payroll.
External Worker ID	worker's identification in Human Resources module	Alphanumeric, max 10 characters. If specified, must match a record in Human Resources and Anthology Payroll. Ignored if Worker ID is specified.
Last Name First Name Middle Name	Worker’s name	Alphanumeric, max 50 characters. Ignored if **Worker ID** is specified. Alphanumeric, max 50 characters. Ignored if **Worker ID** is specified. Alphanumeric, max 50 characters. Ignored if **Worker ID** is specified.
National ID	worker's government-issued identification (e.g.: SIN, SSN, NINO, etc.) *	Alphanumeric, max 30 characters. If specified, must match the National ID on a worker or worker position in Anthology Payroll. Ignored if **Worker ID** is specified.

Position Identification

Position information is typically only for time and earning records. If position info is imported for a benefit/deduction amount, Anthology Payroll attempts to map the benefit/deduction code to a benefit/deduction assigned to the worker at the position level.

Position information is optional. If position information is included, the worker's position must be active for the date of the transaction imported. A worker's dates can be viewed on their **Worker positions** form (**Anthology Payroll** > **Common** > **Anthology Payroll workers** > *select worker* > **Payroll Profile** tab > **Positions**).

Worker position information is stored in the *PHRWorkerPositions* table.

Field	Description	Validation
Position ID	worker's position identifier	Numeric, integers only. Must match a corresponding record in Anthology Payroll and be assigned to the worker indicated and active for the date specified.
Position description	long description of position	Alphanumeric, max 50 characters. If specified, must match position description for one of the positions assigned to the worker indicated. Ignored if Position ID is specified.
Position title	position occupation or type of work	Alphanumeric, max 50 characters. If specified, must match a valid occupation code in Anthology Payroll and be associated to one of the positions assigned to the worker indicated. Ignored if Position ID is specified.
Position filter	jurisdictional information on the time worked, if applicable	Alphanumeric, max 20 characters. If specified, must match a valid position filter in Anthology Payroll. Can be specified as an override to the worker's default position filter on their position record.

Date

Date information is only valid for time and earning records. It is not valid for benefit/deduction records.

There must be at least one piece of information that identifies the date for imported time or earnings. If there is none, an exception is logged.

***Note**: The date must fall within the current pay period or be less than the current pay period. Dates that fall into future pay periods are ignored and not processed.*

Field	Description	Validation
Date	Date for the time/earning record	Alphanumeric, max 10 characters. Accepted formats: • MM/DD/YYYY • MMDDYYYY If the year is not specified, the current year is assumed.
Year	Breakdown of the date	Numeric, integers only, max 4 characters. Must be between 2000 and 2099. Ignored if **Date** is specified.
Month		Numeric, integers only, max 2 characters. Must be between 1 and 12. Ignored if **Date** is specified.
Day		Numeric, integers only, max 2 characters. Must be between 1 and 31. Day number must be valid for the month and year. Ignored if **Date** is specified.

Start Time

Start time information is only valid for time and earning records. It is not valid for benefit/deduction records.

Start time information is ignored if hours or units are present. If neither are present and this is a time or earning record, start time is required.

Field	Description	Validation
Start Time	Worker's shifts start time	Alphanumeric, max 5 characters. Accepted formats: • HH:MM • HHMM HH must be between 0 and 23, where 0 is midnight and 23 is 11pm. MM must be between 0 and 59.
Start Hour	Breakdown of the start time	Numeric, integers only, max 2 characters. Must be between 0 and 23, where 0 is midnight and 23 is 11pm. Ignored if **Start Time** is specified.
Start Minute		Numeric, integers only, max 2 characters. Must be between 0 and 59. Ignored if **Start Time** is specified.

End Time

End time information is only valid for time and earning records. It is not valid for benefit/deduction records.

End time information is ignored if hours or units are specified for the time or earning record. If neither are specified, end time is required.

Field	Description	Validation
End Time	Worker's shift end time	Alphanumeric, max 5 characters. Accepted formats: • HH:MM • HHMM HH must be between 0 and 23, where 0 is midnight and 23 is 11pm. MM must be between 0 and 59.
End Hour	Breakdown of the end time	Numeric, integers only, max 2 characters. Must be between 0 and 23, where 0 is midnight and 23 is 11pm. Ignored if **End Time** is specified.
End Minute		Numeric, integers only, max 2 characters. Must be between 0 and 59. Ignored if **End Time** is specified.

Hours

Hours information is only valid for time and earning records. It is not valid for benefit/deduction records.

Hours information is required unless start and end times, or units are specified for the time or earnings record.

Field	Description	Validation
Hours	Number of hours the worker worked	Numeric, max 13 characters and up to 10 decimal places. Must be between 0.0000000001 and 24.
Whole hours	Integer number of hours the worker worked. Used in conjunction with **Whole minutes**. Ignored if Hours is specified.	Numeric, integers only, max 2 characters. Must be between 0 and 24. Ignored if **Hours** is specified.
Whole minutes	Integer number of minutes. Used in conjunction with Whole hours. Must be greater than zero if	Numeric, integers only, max 2 characters. Must be between 0 and 59.

Field	Description	Validation
	Whole hours is zero. Ignored if Hours is specified.	Must be greater than zero if **Whole hours** is zero. Ignored if **Hours** is specified.
Hours Sign	Indicates whether hours are accumulated or reversed. Only required for earnings imports, not required for time imports. Must be blank, minus sign (-) or positive sign (+). Blank is interpreted as positive.	Character only, max 1 character. Only required for earnings imports, not required for time imports. Must be blank, minus sign (-) or positive sign (+). Blank is interpreted as positive.

Units

Units information is only valid for time and earning records. It is not valid for benefit/deduction records. Units information is optional if start and end times are included.

Unit fields must be empty if hours are specified on the imported time record. You can choose to import units or hours, but not on a single record.

If units are specified, piece code information must also be specified.

Field	Description	Validation
Units	Number of units the worker produced	Numeric, max 20 characters and up to 10 decimal places. Must be between 0.0000000001 and 999999999.9999999999.
Whole units	Integer number of units. Ignored if Units is specified.	Numeric, max 9 characters. Must be between 1 and 999999999. Ignored if **Units** is specified
Units Sign	Indicates whether units are accumulated or reversed	Character only, max 1 character. Must be blank, minus sign (-) or positive sign (+). Blank is interpreted as positive.

Shift

Shift information is only valid for time records and is required. It is not valid for earning or benefit/deduction records.

Shift information is required if hours are imported. If a shift code is imported, its effective date range must include the date for the time record. Shift information cannot be specified together with piece information.

Shift codes are stored in the *PHRShifts* table.

Field	Description	Validation
Shift	shift code the worker worked	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology
Shift description	long description of the shift code	Alphanumeric, max 50 characters. If specified, must match shift description for a shift code in Anthology Payroll. Ignored if Shift is specified.

Earnings

Earnings information is valid only for earnings imports and is required. It is not valid for time or benefit/deduction records.

If an earning code is imported, the effective date range of the earning code must include the date for the earning record.

Earnings codes are stored in the *PHREarns* table.

Field	Description	Validation
Earning	earning code that the worker's earning is charged to	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Earning description	long description of the earning code	Alphanumeric, max 50 characters. If specified, must match earning description for a earning code in Anthology Payroll. Ignored if Earning is specified.

Piece

Piece information is optional, but if unit information is present and this is a time import, then piece information is required. Piece information is not valid for earnings or benefit/deduction imports.

Piece information cannot be specified together with shift information. If a piece code is imported, its effective date range must include the date for the time record.

Piece codes are stored in the *PHRPieces* table.

Field	Description	Validation
Piece	piece code the worker produced	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll
Piece description	long description of the piece code	Alphanumeric, max 50 characters. If specified, must match piece description for a piece code in Anthology Payroll. Ignored if **Piece** is specified.

Amount

Earning amount information is valid only for earnings imports. It is not valid for time or benefit/deduction imports.

Earning amount information is required, unless the earning is hours-based and hours are specified. Earning amount can be zero. The currency is assumed to be the default currency of the pay group.

Field	Description	Validation
Amount	total amount of the earning	Numeric, max 12 characters and up to 2 decimal places.
Amount Sign	negative or positive indicator of the amount	Character only, max 1 character. Only required for earnings imports, not required for time imports. Must be blank, minus sign (-) or positive sign (+). Blank is interpreted as positive.
Rate	worker's hourly rate of pay	Numeric, max 20 characters and up to 10 decimal places. Ignored if **Amount** is specified.

Salary

Salary information is optional. Salary information is only valid for earnings imports. It is not valid for time or benefit/deduction imports.

If a salary grid is imported, its effective date range must include the date for the earning record.

Salary grid codes are stored in the *PHRSlryGrids* table. Salary steps are stored in the *PHRSlrySteps* table.

Field	Description	Validation
Salary Grid	Salary grid code used to derive an hourly rate	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Salary Grid Description	Long description of the salary grid	Alphanumeric, max 50 characters. If specified, must match salary grid description for a salary grid in Anthology Payroll. Ignored if **Salary Grid** is specified.
Salary Step Number	Worker's current salary step number within the salary grid used to derive the hourly rate	Numeric, integers only, max 10 characters. This field is required if **Salary Grid** or **Salary Grid Description** are specified.

Ledger

Ledger information is optional, unless dimension information is specified, in which case ledger information is required. Ledger information is only valid for earnings imports. If specified, it must exist in the General Ledger.

Field	Description	Validation
Ledger account	General ledger code that the earning is charged to	Alphanumeric, max 10 characters. Must match a corresponding entry in the General Ledger.
Ledger description	Long description of the ledger account	Alphanumeric, max 60 characters. If specified, must match ledger description for a ledger account in General Ledger. Ignored if **Ledger** is specified.

Department

Department information is optional and describe specific organization units that time and earnings can be charged to. Department information is only valid for time and earnings imports.

Field	Description	Validation
Organization unit	department unit code that the earning is charged to	Alphanumeric, max 20 characters. Must match a corresponding record in Organization Administration.
Organization description	long description of the department unit	Alphanumeric, max 50 characters. If specified, must match department description for a department in Organization Administration. Ignored if Organization unit is specified.

Dimension

Dimension information is optional and is used to identify an organization unit. Dimension information is only valid for time and earnings imports. If dimension information is specified, ledger account information must also be specified.

If the dimension fields cannot uniquely identify an organization unit, the time or earnings record is not imported.

Dimension information is ignored if **Organization unit** is specified.

Field	Description	Validation
Dimension 1	additional financial dimensions for the department (organization unit). Financial dimensions are ordered alphabetically in Microsoft Dynamics 365 Finance. The order that they are included in the import file must be the same alphabetical order that they are defined in Dynamics. Please note that this order can change if you later add more dimensions as Dynamics reorders them alphabetically.	Alphanumeric, max 10 characters.
Dimension 2		
Dimension 3		
Dimension 4		
Dimension 5		
Dimension 6		
Dimension 7		
Dimension 8		
Dimension 9		
Dimension 10		

Occupation

Occupation information is optional. Occupation information is only valid for time and earnings imports.

Occupation information is optional. If occupation is imported, its effective date range must include the date for the time or earning record.

Occupation codes are stored in the *PHROccupations* table.

Field	Description	Validation
occupation	occupation associated with the worker's position	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
occupation description	long description of the occupation	Alphanumeric, max 50 characters. If specified, must match occupation description for an occupation in Anthology Payroll. Ignored if **Occupation** is specified.

Job

Job information is optional. Job information is only valid for time and earnings imports.

If a job is imported, its effective date range must include the date for the time or earning record. Job codes are stored in the *HcmJob* table.

Field	Description	Validation
Job	job associated with the worker's position	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Job description	long description of the occupation	Alphanumeric, max 50 characters. If specified, must match occupation description for an occupation in Anthology Payroll. Ignored if **Job** is specified.

Project

Project information is optional. Project information is only valid for time and earnings imports.

Note: The **Project** and **Category** parameters are critical to the Anthology Payroll integration with Project. They determine whether and how actual Anthology Payroll expenses are sent to Project.

Field	Description	Validation
Task	project task associated with the earning	Alphanumeric, max 20 characters. Must match a corresponding record in Advanced Payroll.
Task description	long description of project task	Alphanumeric, max 50 characters. If specified, must match task description for a task in Anthology Payroll. Ignored if Task is specified.
Project	identifier in Project Management and Accounting module for the project that time is charged to	Alphanumeric, max 10 characters. Must match a project identifier in the Project Management and Accounting module.
Category	project category associated with the earning	Alphanumeric, max 10 characters. Must match a category identifier in the Project Management and Accounting module.
Activity	project activity number associated with the earning	Alphanumeric, max 50 characters. Must match an activity number in the Project Management and Accounting module.
Journal	project journal for standard cost accrual associated with the earning	Alphanumeric, max 10 characters. Must match a journal identifier in the Project Management and Accounting module
Line	line number within the project journal associated with the earning	Numeric, max 10 characters. Must match a line number in the Project Management and Accounting module.
Reversal category	project reversal category, if earning amount is to be reversed	Alphanumeric, max 10 characters. Must match a reversal category identifier in the Project Management and Accounting module.

Reasons

Reason information is optional. Reason information is only valid for time and earnings imports. If a reason is imported, its effective date range must include the date for the time record.

Reason codes are stored in the *PHRReasons* table.

Field	Description	Validation
Work reason	Work reason encoded carried forward from time transaction, if applicable	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Work reason description	Long description of work reason	Alphanumeric, max 50 characters. If specified, must match reason description for a reason in Anthology Payroll. Ignored if **Work reason** is specified.
Relief reason	Reason for working in relief of another worker, if applicable. Carried over from time transaction.	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Relief description	Long description of relief reason	Alphanumeric, max 50 characters. If specified, must match reason description for a reason in Anthology Payroll. Ignored if **Relief reason** is specified.

Benefit/deduction Amount

Benefit/deduction amount information is only valid for benefit/deduction imports.

Benefit/deduction amount can be imported as a flat amount or as a calculation, but not both. The amount can be specified numeric amounts or calculation codes. The currency is assumed to be the default currency of the pay group.

Field	Description	Validation
Worker Amount	Benefit/deduction amount paid by or deducted from the worker	Numeric, max 12 characters and up to 2 decimal places.
Worker Amount Sign	Negative or positive indicator of the worker amount	Character only, max 1 character. Must be blank, minus sign (-) or positive sign (+). Blank is interpreted as positive.
Employer Amount	Benefit/deduction amount paid by the employer on the worker's behalf	Numeric, max 12 characters and up to 2 decimal places.

Field	Description	Validation
Employer Amount Sign	Negative or positive indicator of the employer amount	Character only, max 1 character. Must be blank, minus sign (-) or positive sign (+). Blank is interpreted as positive.
Worker Calculation	Calculated amount that the worker pays for this benefit or deduction	Alphanumeric, max 20 characters. Must match a corresponding record in Advanced Payroll.
Worker Calculation Description	Long description of the specified worker calculation	Alphanumeric, max 50 characters. If specified, must match calculation description for a calculation code in Anthology Payroll. Ignored if **Worker Calculation** is specified.
Employer Calculation	Calculated amount that the employer pays on the worker's behalf for this benefit or deduction	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Employer Calculation Description	Long description of the specified employer calculation	Alphanumeric, max 50 characters. If specified, must match calculation description for a calculation code in Anthology Payroll. Ignored if **Employer Calculation** is specified.
Payment Type	Type of payment on which the benefit/deduction can appear	Alphanumeric, max 20 characters. Must match a corresponding record in Anthology Payroll.
Payment Type Description	Long description of the payment type	Alphanumeric, max 50 characters. If specified, must match payment type description for a payment type code in Anthology Payroll. Ignored if **Payment Type** is specified.

Coding Samples

The following examples show sample code for performing standard functions with the Anthology Payroll API.

Add Time or Earnings

```
PHRImport importEarn; str fields;

container values;

void getExceptions(PHRImport import)

{

numOfExceptions = import.getNumOfExceptions(); for(i = 1 ; i <=
numOfExceptions; ++i)

{

info(import.getExceptions(i));

}

}

;

try

//fields = 'worker id, date, earning, amount'; // use these fields
for earnings. These are just examples, refer to specification for
additional fields

//fields = 'worker id, date, shift, hours'; // use these fields for
time. These are just examples, refer to specification for additional
fields

{

importEarn = PHRImport::construct(PHRImportType::Earn) //For time,
use PHRImportType::Time

payGrp = PHRImport::getPayGroup(HcmWorkerRecId, HcmPositionRecId,
1/1/2020);


// initialize the importEarn

values = ["000456", 1/1/2020, "Regular",8] //these fields can be for
earnings or time, assuming "Regular" refers to a shift code or
hourly earning code

if(!importEarn.init(payGrp, fields))

{

getExceptions(importEarn);

}
```

```
run.

 // validate entries or validate/commit entries

// only one of importValidate method or importCommit method should
be


// All imported transactions can be viewed and committed to
transaction


tables from the appropriate import journals in the Dynamics client
application.

for(i = 1; i<=conlen(values); ++i)

{

// validate the entry if(!importEarn.importValidate(conpeek(values,
i)))

{

getExceptions(importEarn);

throw error("exception occurred");

}

}

// This method validates, imports and commits the transactions from
the container object to live data in. This does everything that
importValidate method does as well as committing the transactions to
live transaction tables. This method is optional. You may choose to
use importValidate to import transactions and commit from the client
application use importCommit to do all those tasks from the API.

//if (!importEarn.importCommit(conpeek(values, i)))

//{

// getExceptions(importEarn);

// throw error("exception occurred");

//}

}

// Must call this method importEarn.importFinalize();

//////////////////////////////////////////////////////
PHRInfo::showInfo('Import process finished');

{

catch(exception::Error)
```

```
}
PHRInfo::showError('Import process aborted due to errors.');
{
//Must call this method importEarn.importFinalize();
```

Frequently Asked Questions

Question: I need to create one import object per pay group. I do not know how many pay groups there are prior to running the import. It would usually be only one, but there could be more than one. How do I determine how many objects to declare?

Answer: Create objects dynamically and maintain them in a list or a map.

Question: Do you have a method that I can use to retrieve a worker's position based on: HcmWorkerRecId, PHRUNIONAFFLNCODE, PHRUNIONCERTCODE, and PHROCCUPATIONCODE? I need to locate the position with these criteria so that I can send the correct Position to the Anthology Payroll Time journal. I also need to be able to determine what the worker's primary position is so that I can default in the PHRUnionAfflnCode on the time capture screen. What would be the best way to determine the primary position?

Answer: Rather than providing these as search criteria, iterate through the returned container contents, since there will likely be very few positions returned for a particular worker. An alternative would be just to use the find method on PHRWorkerPositions, the signature of which is:

```
static PHRWorkerPositions find(HcmWorkerRecId _worker,
HcmPositionRecId _position, PHRDate _contextDate = systemdateget(),
boolean _forUpdate = false, PHRSelectType _selectType =
PHRSelectTypeValues::EffectiveDated)
```

Use:

```
PHRWorkerPositions::find(worker, position, contextDate);
```

The find method can be used in other worker tables to get a specific record.

Question: Can you tell me the best way to determine the current pay period? I need to validate the Date Worked on the time capture to ensure it is within the current pay period.

Answer: To determine the current pay period:

```
static void currentPayPeriod(Args _args)
{
    PHRDatePeriodDates datePeriodDates;
    PHRPayGrpCode payGrpCode;
    PHRDatePeriodStartDate datePeriodStartDate;
    PHRCurrentPayPeriodEndDate currentPayPeriodEndDate;
    PHRDatePeriodCode datePeriodCode;
```

```
    PHRDatePeriodNum datePeriodNum;
    PHRDatePeriodYear datePeriodYear;
    ;
    payGrpCode = "Staff";
    datePeriodDates = PHRDatePeriodDates::findByPayGrp(payGrpCode);
    currentPayPeriodEndDate = datePeriodDates.DatePeriodEndDate;
    datePeriodStartDate = datePeriodDates.DatePeriodStartDate;
    datePeriodCode = datePeriodDates.DatePeriodCode;
    datePeriodNum = datePeriodDates.DatePeriodNum;
    datePeriodYear = datePeriodDates.DatePeriodYear;

    info(strfmt("datePeriodCode : %1, datePeriodNum : %2,
datePeriodYear : %3, datePeriodStartDate : %4,
currentPayPeriodEndDate : %5", datePeriodCode, datePeriodNum,
datePeriodYear, datePeriodStartDate, currentPayPeriodEndDate));
}
```

Question: We are using the Anthology Payroll field PHRWorkerId rather than the HR WorkerID field on the time capture form to ensure that when they enter time, they are selecting employees that are in Anthology Payroll. How do I find their PHRWorkerId's?

Answer: Class PHRLookup Method:

```
client static void lookupWorkerID(FormControl _formControl, date
_from = systemdateget(), date _to = _from, PHRPayGrpCode _payGrpCode
= '', boolean _isExpired = false, boolean _isActive = true, boolean
_isFuture = false, boolean _showDates = false)
```

Use:

```
PHRLookup::lookupWorkerID(this, PayPeriodStartDate,
PayPeriodEndDate, PayGrpCode);
```

Question: If my import of time is successful, I do the commit and finalize on the PHRImportTime object, and then I proceed to process earnings. My problem is that if the earnings import fails, then the Time has already been committed and it is not rolled back. The earnings roll back and so do all of my project journals, but not the Anthology Payroll time.

Answer: Don't commit time until after earnings are processed, too. Commit both at the end.

Affected Tables

Table	Description
DirPersonName	Contains names of workers
HcmWorker	Contains identification numbers of workers
PHRBenDeds	Contains benefit/deduction codes
PHREarns	Contains earning codes
PHROccupations	Contains the occupations for which time or earnings can be recorded
PHROrgs	Contains the organization levels against which time or earnings can be recorded
PHRPayGrps	Contains pay group codes
PHRPieces	Contains piece codes
PHRReasons	Contains the reasons against which a worker's time or earnings can be recorded
PHRShifts	Contains shift codes
PHRSlryGrids	Contains salary grids for deriving hourly rates
PHRSlrySteps	Contains salary steps for deriving hourly rates
PHRWorkerBenDedAmtImports	Contains benefit/deduction information for workers imported from external sources
PHRWorkerBenDedScheds	Contains scheduled benefit/deduction information for workers
PHRWorkerEarnImports	Contains earnings information for workers imported from external sources
PHRWorkerEarns	Contains earnings information for workers
PHRWorkerPositions	Contains the worker position information and the positions for which a worker's time or earnings can be recorded
PHRWorkers	Contains worker information
PHRWorkerTime	Contains the time information for workers
PHRWorkerTimeImports	Contains time information for workers imported from external sources