History

History of changes monitors and records changes made to the records in the database during insert, update and delete. It is capable of restoring either the database or the record into its previous state.

Introduction

The History monitors all operations with the database. It is incorporated into the Database Layer. You can find it is called in every version of methods **executeInsert**, **executeUpdate**, **executeDelete** except those that contain the suffix History.

Thus the History is usually saved when working with the Application. There are several cases when History is not saved

1. During Table Import - data in the Immutable tables TPlants, TVillage, TPhytochoria, and TTerritoriess are modified without saving the changes; it is the Table Import that should be used to update the contents of those tables.
2. When adding/editing Users of the database.

The following table shows the list of tables and columns for which the History is recorded.

Table	***Columns***
tOccurrences	cYearCollected, cMontCollected, cDayCollected, cTimeCollected, cDataSource, cHerbarium, cNote, cHabitatId, cPlantId, cPublicationId, cMetadataId
tAuthorsOccurrences	cRole, cNote
tAuthors	cWholeName, cOrganization, cTelephoneNumber, cRole, cAddress, cEmail, cURL, cNote,
tHabitats	cQuadrant, cDescription, cCountry, cAltitude, cLatitude, cLongitude, cNote, cTerritoryId, cPhytochoriaId, cNearestVillageId, tPhytochoria, tVillages, tTerritories, tPublications, tAuthor, tMetadata
tMetadata	cTechnicalContactName, cTechnicalContactEmail, cTechnicalContactAddress, cContentContactAddress, cContentContactEmail, cContentContactAddress, cDataSetDetails, cDatasetTitle, cSourceInstitutionId, cSourceId, cOwnerOrganizationAbbrev, cRecordBasis, cBiotopeText

The changes are stored right into the databas. There are three tables that are used for it - they are tHistory, tHistoryChange and tHistoryColumn. The detailed description of these tables can be found the the Section **Database Model**. In short:

1. tHistoryColumn is a table with fixed contents. It contains pairs <Table, Column> for each table and column for which the History is recorded.
2. tHistoryChange contains details about the record that was changed (cRecordID) - the type of operation (cOperation), the date and time of the operation (cWhen), and the User who performed the operation (cWho).
3. tHistory contains information about the column that was changed (the cColumnId is a foreign key to the tHistoryColumn table - identifying the name of the table and the name of the column that was changed) and contains both the new value and old value in that column (cOldValue and cNewValue). In order to know when the change occurred and who made it to which record, there is a reference to the record in the tHistoryChange (cChangedId).

Participants

There are two participants this time. The first one is the Database Layer itself that records the changes in all its methods starting with the prefix `**execute**` that do not have the suffix `**History**`.

The second one is History.java, that can restore the state of records in the database.

We will describe how exactly the History is saved.

Hibernate Database Layer

The Database Layer stores changes to the tHistory and tHistoryChange tables. We will discuss the values that can be in their columns.

The source is in the file net.sf.plantlore.server.HibernateDBLayer.java.

These columns **always** contain the same values:

Table.Column	*Value*
tHistoryChange.cWho	The User that performed the operation
tHistoryChange.cWhen	The date and time of the operation
tHistoryChange.cOperation	1 = insert; 2 = update; 3 = delete
tHistory.cChangeId	The cID of the record in the tHistoryChange that describes the record that has been modified (its cID, when did the change happen, who made the change, and the type of the change).

Insert

These columns always contain the same values when inserting a new Record:

Table.Column	*Value*
tHistoryChange.cRecordId	The cID of the inserted record.
tHistory.cColumnId	The cID of the record in the tHistoryColumn that satisfies: cTable = Record & cColumn = null (e.g. cTable = Habitat & cColumn = null)
tHistory.cOldRecordId	0
tHistory.cOldValue	null
tHistory.cNewValue	null

The only exception is the Insert of a new record into the tAuthorsOccurences table.

1. In case the insert occurs when inserting a new Occurrence, the History should not be recorded - **executeInsertHistory()** or **executeInsertInTransactionHistory()** should be used!
2. In case the insert occurs when adding a new AuthorOccurrence to an already existing Occurrence, the History should be recorded - **executeInsert()** or **executeInsertInTransaction()** should be used. Besides tHistoriChange.cOperation = 2.

Update

These columns always contain the same values when updating an existing Record:

Table.Column	*Value*
tHistoryChange.cRecordId	The cID of the updated record.
tHistory.cColumnId	The cID of the record in the tHistoryColumn that satisfies:

Table.Column	*Value*
	cTable = Record & cColumn = Name of the updated column. (e.g. cTable = Habitat & cColumn = Description)
tHistory.cOldRecordId	If the Record contains some foreign keys and the foreign key was updated to another foreign key, the original value of the foreign key is stored. Otherwise this column is 0.
tHistory.cOldValue	The old value contained in the column.
tHistory.cNewValue	The new value that replaced the old one.

Delete

These columns always contain the same values when deleting an existing Record:

Table.Column	*Value*
tHistoryChange.cRecordId	The cID of the deleted record.
tHistory.cColumnId	The cID of the record in the tHistoryColumn that satisfies: cTable = Record & cColumn = null (e.g. cTable = Habitat & cColumn = null)
tHistory.cOldRecordId	0
tHistory.cOldValue	null
tHistory.cNewValue	null

Please note that deletion of AuthorOccurrences is not recorded by History, at least not directly. In case an Occurrence is marked as deleted (i.e. Occurrence.cDelete = 1), all associated AuthorOccurrences that have their cDelete = 0 should be marked as deleted but with the AuthorOccurrence.cDelete = 2. This way if the Occurrence is undeleted, it would be clear which AuthorOccurrences should be undeleted as well.

History

History can undo the changes made to the database. Most important methods are **undoToDate()** and **undoSelected()**. Because the list of stored changes can become very large, History provides methods to dispose of the whole History - **clearDatabase()**.

The source is in the file net.sf.plantlore.client.history.History.java.