# Changelog Behavior

- Enabling Changelogs for a Model
- What ChangelogBehavior Does
  - Save (Add)
  - Save (Update)
  - Delete
  - Find
- Sample Relations

COmanage Registry can track each change made to a Model through the use of Changelogs. When a record is modified, the old values are archived in order to provide an audit or change log of the modifications that were made. Changelogs are implemented using a custom Cake Behavior (ChangelogBehavior).

Changelog Behavior operates using a modified *copy on write* design. When a record is changed, the old values are copied to a new record, and then the original record is updated with the new values. There are two reasons for this design: it avoids massive rekeying when a record is updated (otherwise related Models would all need to be re-keyed to point to the new record), and because it's tricky within Cake to rewrite the record ID in the middle of an operation. (That is, for `/foos/edit/23`, Cake really wants the item being edited to be #23, and not some new number.) It is simply easier to make the archived copy the new record.

This behavior can be modified using the model variable `$relinkToArchive`. If a Model defines a related *hasOne* or *hasMany* Model in this variable, then ChangelogBehavior will relink the related model to the archive copy, as part of the `afterSave` callback. This is useful for maintaining historical context, such as for Petition attribute definitions. If the Model relationship is *belongsTo*, a similar rewrite happens, but in `beforeSave` and with the assumption that the parent model is what was changed. (The *belongsTo* relationship is used for when related Models are saved simultaneously as part of a *saveAll* or *save Associated* call.)

See also: Understanding Registry Deletion

## Enabling Changelogs for a Model

**Behavior Priority**

ChangelogBehavior *must* execute before ContainableBehavior in order to properly filter results from contain'd finds. The default priority for behaviors is 10, so setting the priority to 5 will cause the ChangelogBehavior to execute first.

NormalizationBehavior *should* execute before ChangelogBehavior in order to clean up data before any save happens.

As with other Behaviors, the appropriate Model should define something like

```
$actsAs = ("Changelog" => array('priority' => 5));
```

In order to support Changelogs, a Changelog-enabled Model must define several fields. As described below, these fields are used to track changed records:

- `revision`
- `deleted`
- `actor_identifier`
- *`model`*`_id`

For performance reasons, *model_id* should be indexed.

`$relinkToArchive` should also be defined if needed, as described above.

```
class Model1 extends AppModel {
  public $hasMany = array('Model2');
  public $relinkToArchive = array('Model2');
}
```

When a hard delete must be performed, ChangelogBehavior can be initialized with the setting *expunge* set to true. This setting is also supported by StandardController, via the controller level setting `$useHardDelete`.

```
$model->reloadBehavior('Changelog', array('expunge' => true));


class ModelController extends StandardController {
  public $useHardDelete = true;
}
```

**About actor_identifier**

*actor_identifier* stores the username (identifier) of whoever made the change. This is a string rather than a foreign key into cm_co_people for several reasons:

- No need for foreign keys and Model relations, which must be added to each Changelog model, and are tricky to automatically create for tables created in `schema.xml` before *cm_co_people*.
- Audit integrity is maintained even after an expunge.
- Some models are not attached to a CO, so tracking a CO Person as an actor doesn't quite make sense.

**Extended Attributes**

Extended Attributes are handled differently. They are treated as "extensions" of CoPersonRole, and archived copies are manually maintained by CoPersonRolesController.

## What ChangelogBehavior Does

ChangelogBehavior intercepts *find*, *save*, and *delete* requests to transparently manage the Changelog records.

### Save (Add)

- `revision` is set to 0.
- `actor_identifier` is set to the current username.

### Save (Update)

- The prior values are copied to a new row, and `model_id` is set to point to the parent record.
- `revision` is incremented.
- If the Model defines related models in `$relinkToArchive`, those related models will be re-keyed so that the related records point to the archived attribute instead of the current attribute. (By default, related models remain linked to the current attribute.)

### Delete

- `deleted` is set to true (effecting a soft delete), unless ChangelogBehavior is initialized with the setting *expunge* set to true (in which case a hard delete is performed).

### Find

- By default, a find that does not request a specific record ID will be modified to return the most current revision and to exclude deleted records.
- A find with the `changelog` parameter `archived` set to true will return all records, not just those described above.

  ```
  $args['conditions']['Model.field'] = $requestedField;
  $args['changelog']['archived'] = true;

  $Model->find('all', $args);
  ```

- By passing the `changelog` parameter `revision`, a find can request a specific revision for a record:

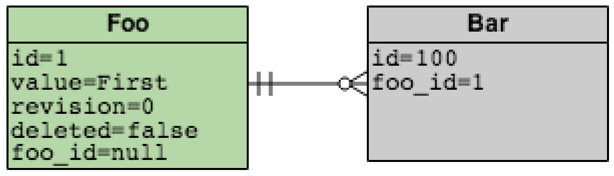
  ```
  $args['conditions']['Model.id'] = $requestedId;
  $args['changelog']['revision'] = $requestedRevision;

  $Model->find('first', $args);
  ```

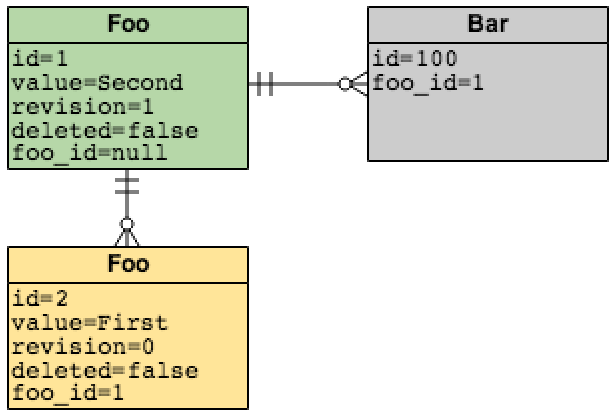
- If ChangelogBehavior is initialized with the setting *expunge* set to true, all records will be returned, not just those described above.

Sample Relations

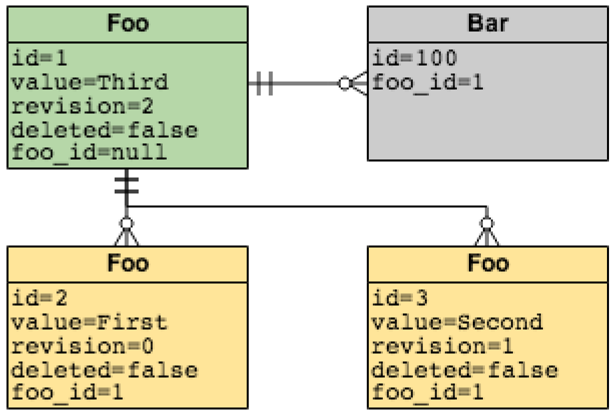
*Add Foo, which HasMany Bar*



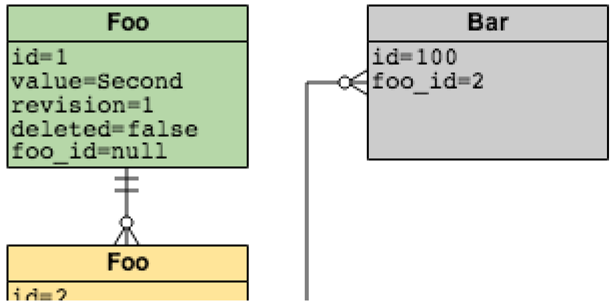
*Update Foo*

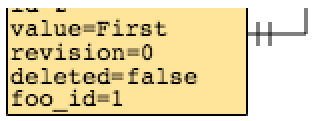


*Update Foo again*

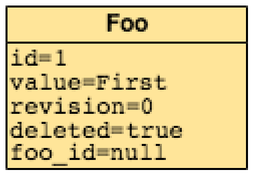


*Update Foo, but with relinkToArchive*




value=First
revision=0
deleted=false
foo_id=1

*Delete Foo*


Foo
id=1
value=First
revision=0
deleted=true
foo_id=null