EMu Documentation

Record Level Security

Document Version 1

**EMu version 4.3**

EMu
Museum
Management
System

Contents

Section 1

Record Level Security

Record Level Security provides organizations with control over who can do what to records on a per user and per group basis.

At its simplest it is possible to specify who can view, edit and delete individual records. For instance, it is possible to specify that Everyone can view all Parties records but only managers of each department are able to edit and delete the records of staff members in their department. In the following example, Everyone is able to view the current record (they have the `Display` permission), but only members of group Registrations are able to edit and delete this record (the `Edit` and `Delete` permissions are enabled for group Registrations but disabled for group Everyone):
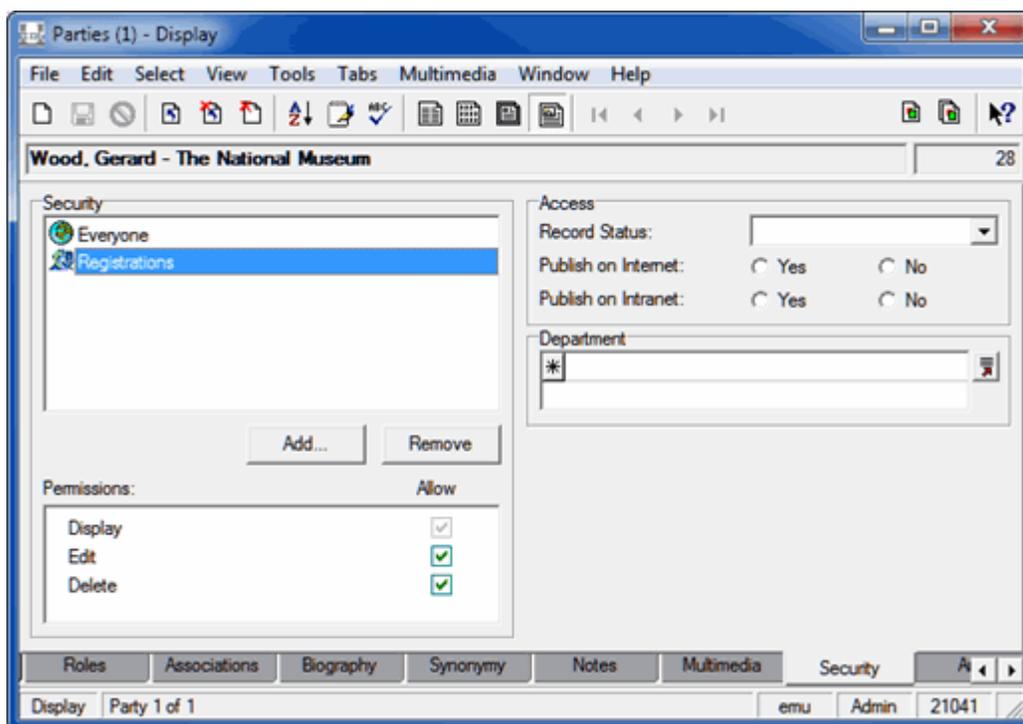
With the Security Registry entry (page 21) however it is possible to manage permissions dynamically so that a user / group's `Display`, `Edit` and `Delete` permissions for a record are conditional upon a value entered in a field (any field) in the module.

In the example above, we have manually:

- Changed the permissions of group Everyone, allowing members to `Display` the record but not to `Edit` or `Delete` it.

-AND-

- Added the Registrations group to the *Security* box, providing members with `Edit` and `Delete` permissions to this record.

With the Security Registry entry it is possible to specify that:

- Members of group Registrations are only able to edit and delete a record if the *Department* field holds the value `Registrations` (in other words, they can only edit and delete their own records)
  -AND-
- When members of group Registrations add a new record:
    - Permissions for group Everyone are limited to `Display`
    - Permissions for group Registrations are set to `Display`, `Edit` and `Delete`
    - The *Department* field is populated with the value `Registrations`

In this way, whenever the value in the *Department* field is updated to hold the value `Registrations` (whether manually or when a new record is added by members of group Registrations), all users will be able to view the record but only members of group Registrations will be able to edit and delete it.

Another useful example of the dynamism inherent to the Security Registry entry is to control who can view, edit and / or delete a record based on a *Record Status* for instance. If *Record Status* changes from, say, `Active` to `Retired`, permissions can be changed dynamically to hide the record from certain groups of users.

> Any field in a module can be used to set conditions when applying Record Level Security. See How to refine Record Level Security by specifying conditional criteria (page 16) for details about refining the three standard security permissions (`Display`, `Edit`, `Delete`).

It is also possible to search for records based on the Record Level Security permissions assigned to users and groups. If a user or group has been removed from EMu, it is still possible to locate records for which they had permissions assigned by using the *Security (Direct)* fields, which are available in Search mode.

What is Record Level Security?

With Record Level Security it is possible to set permissions to control who can:

1. View (Display) a record
2. Edit a record
3. Delete a record

> Keep in mind that Record Level Security permissions sit above the base Operations permissions assigned to groups: even if a group is assigned a Record Level Security `Edit` permission to a record, if the group does not have the `daEdit` Operations permission, they will be unable to edit any record.

Security settings can be set on:

- A per user basis: User A can view but not edit a record for instance.
- A per group basis: Group A can View, Edit and Delete a record.
- On one record at a time.
- On multiple records at a time using the Set Record Security (page 10) batch update tool.

Record Level Security is available in all modules except Field Level Help and is applied on a module's Security tab:
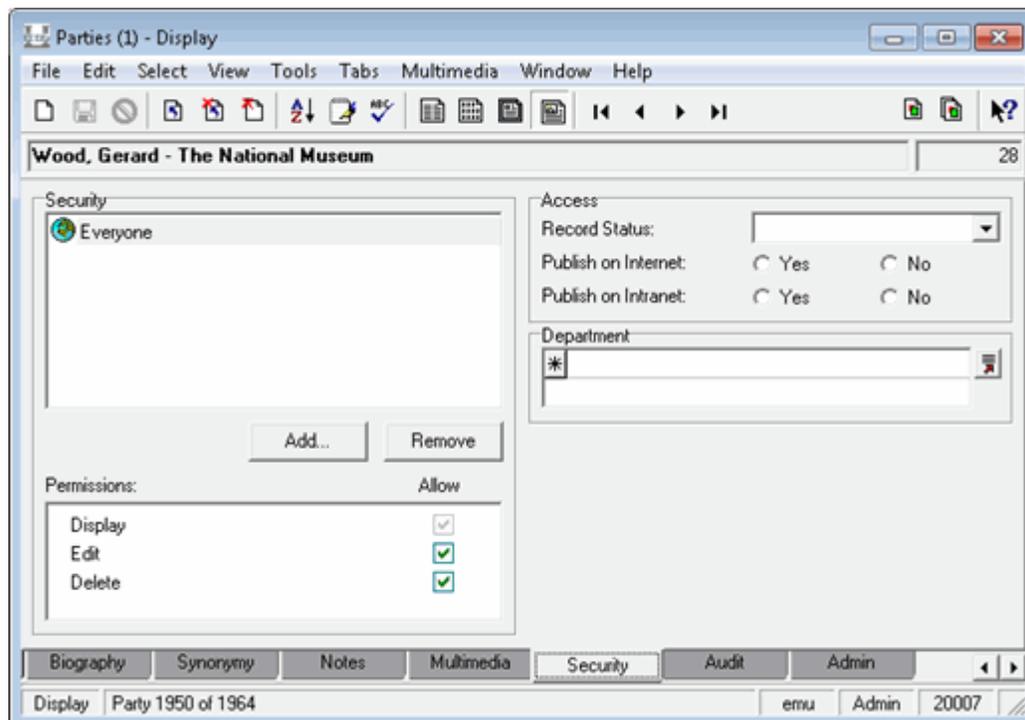
Applying security settings to a record is a simple matter of:

4.   Searching for the record.
5.   Adding or removing a user or group from the *Security* box on the Security tab.
6.   Ticking / unticking the appropriate permissions in the *Permissions* box.

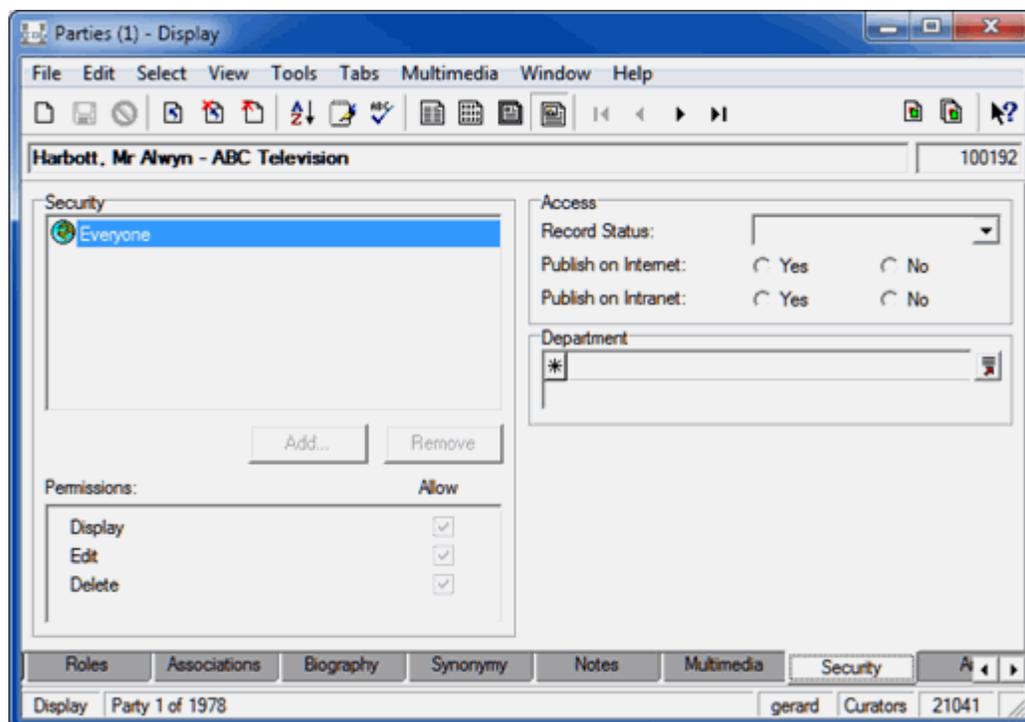In the example above, group Everyone can `Display`, `Edit` and `Delete` this record.

> The minimum permission for a user / group is `Display`: in practice this means that when a user / group is added to the *Security* box, the `Display` checkbox is grayed out and uneditable. To remove all permissions for a user / group, including `Display`, remove the user / group from the *Security* box. As we'll see however (page 12), users inherit permissions from the groups to which they belong. All users, for instance, are members of group Everyone: if group Everyone is added to the *Security* box and it has `Edit` permission enabled, then all users inherit the `Edit` permission for that record.

Who can change Record Level Security settings?

As well as requiring the usual permissions that allow a user to edit records (`daEdit` for instance), two Security conditions must be met before a user can change Record Level Security settings for a record. The user must have (or be a member of a group that has):

1. The `daSecurity` Operations permission.
2. `Edit` permission for the record (the `Edit` checkbox in the *Permissions* box must be ticked).

If a user does not have both of these permissions, the options in the *Security* box will be grayed out and uneditable:

How to set Record Level Security permissions on a record

For this demonstration we permit all users to view (`Display`) a particular record, but only members of group Admin to `Edit` and `Delete` it.
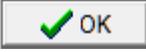
> If you are unable to change the security settings on the Security tab, check that you have the appropriate permissions (page 5) to do so.

1. Search for the record to which to apply permissions.
2. On the Security tab, select group **Everyone** in the *Security* box.
3. Uncheck the `Edit` and `Delete` checkboxes in the *Permissions* box:
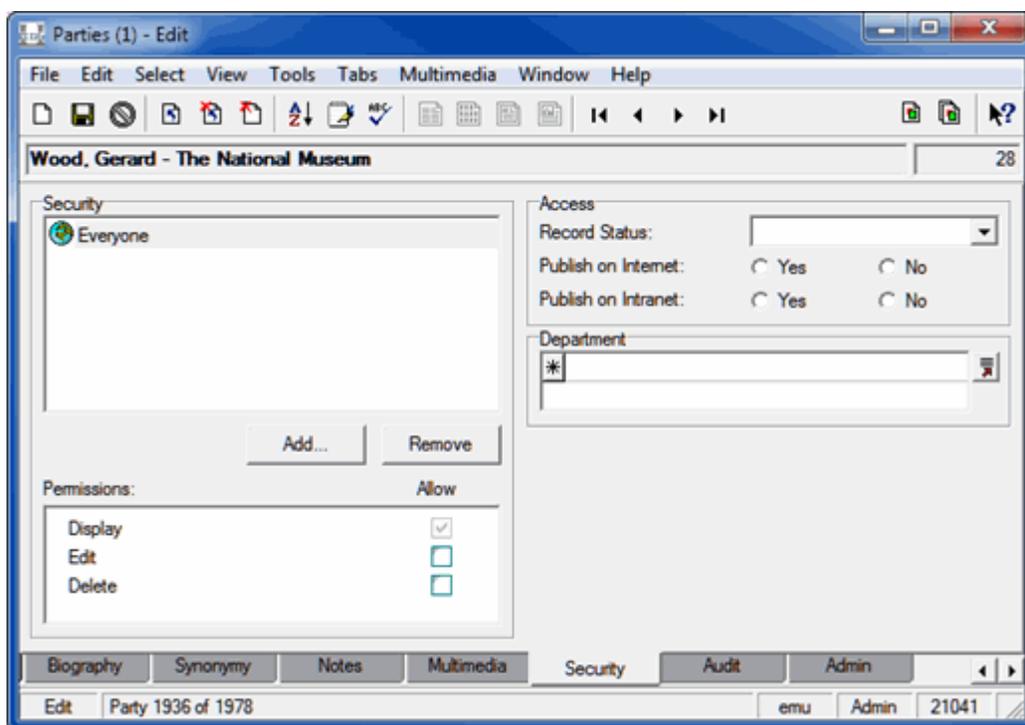
> The minimum permission for a user / group is `Display`: in practice this means that when a user / group is added to the *Security* box, the `Display` checkbox is grayed out and uneditable. To remove all permissions for a user / group, including `Display`, simply remove the user / group from the *Security* box.

4. Click [Add...] to add a user / group.
   The Select User/Group box displays with a list of all available users and groups.

5. Scroll through the list of users / groups and double-click the user / group to be added (or select the user / group and click ✔ OK).
   The user / group (Admin in this example) is added to the *Security* box:
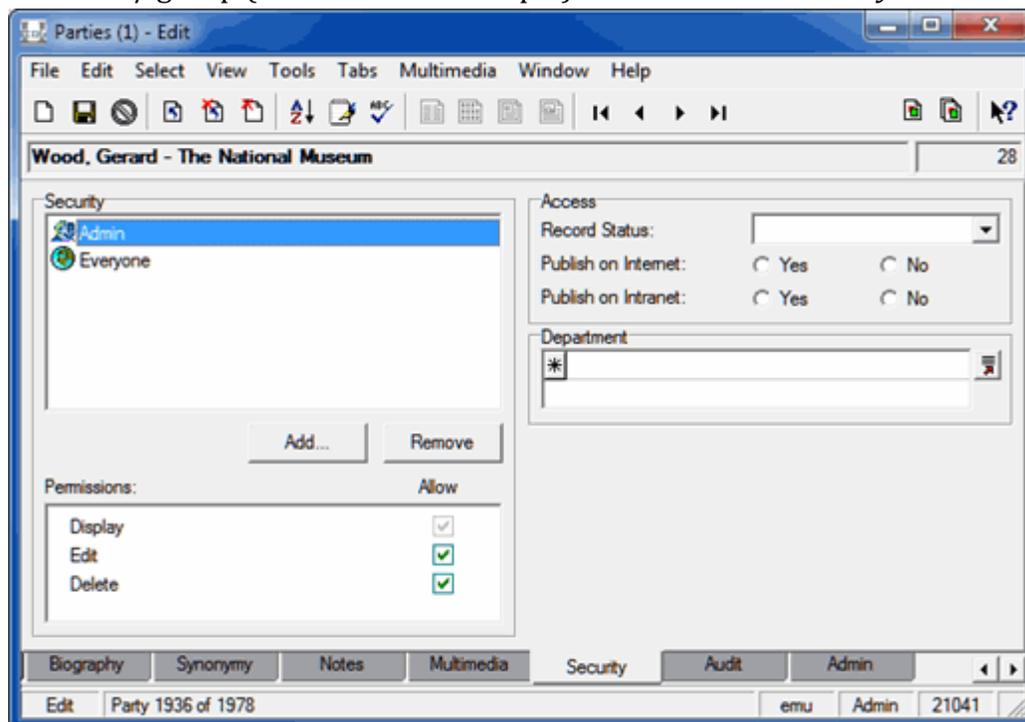
The minimum permission for a group added to the *Security* box is Display and by default this permission is checked for group Admin (in fact, group Admin has inherited this permission from group Everyone). The Edit and Delete permissions can be ticked / unticked for group Admin however as these permissions were not inherited from group Everyone.

Users inherit permissions from groups

Although we look at this issue in more detail elsewhere (page 12), it is worth stressing here that users inherit permissions from the groups to which they belong.

A simple illustration will suffice for now. As the name suggests, all users are members of group Everyone. Let us give Everyone permission to `Edit` this record:
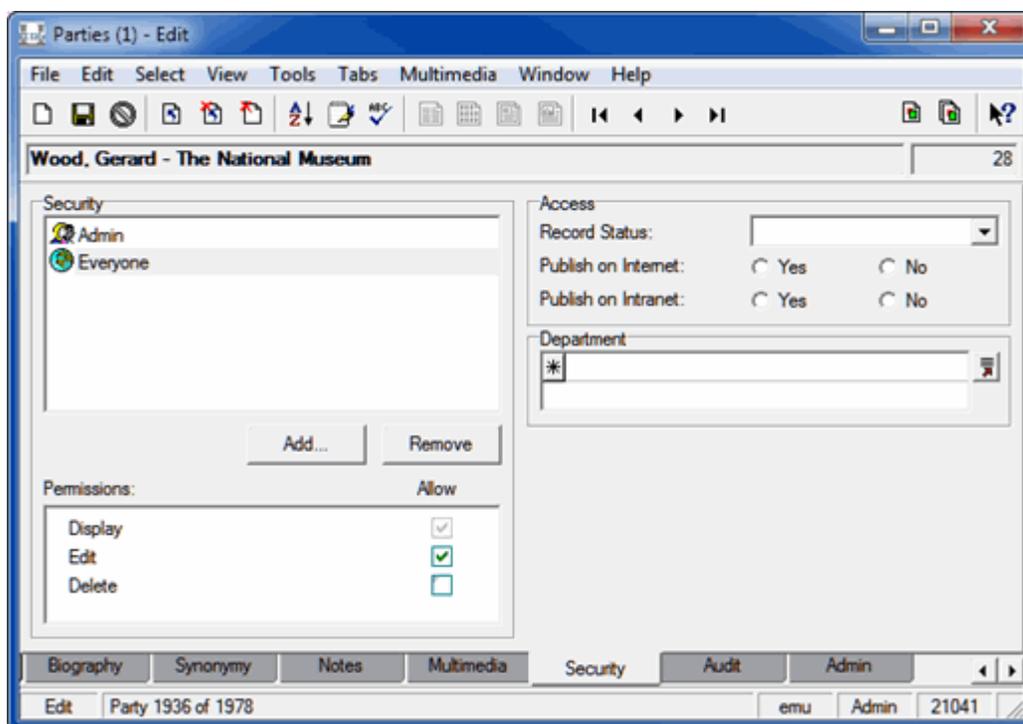
Now when we check the permissions of the Admin group we see that not only is the `Display` permission grayed out and uneditable, but so is `Edit`:
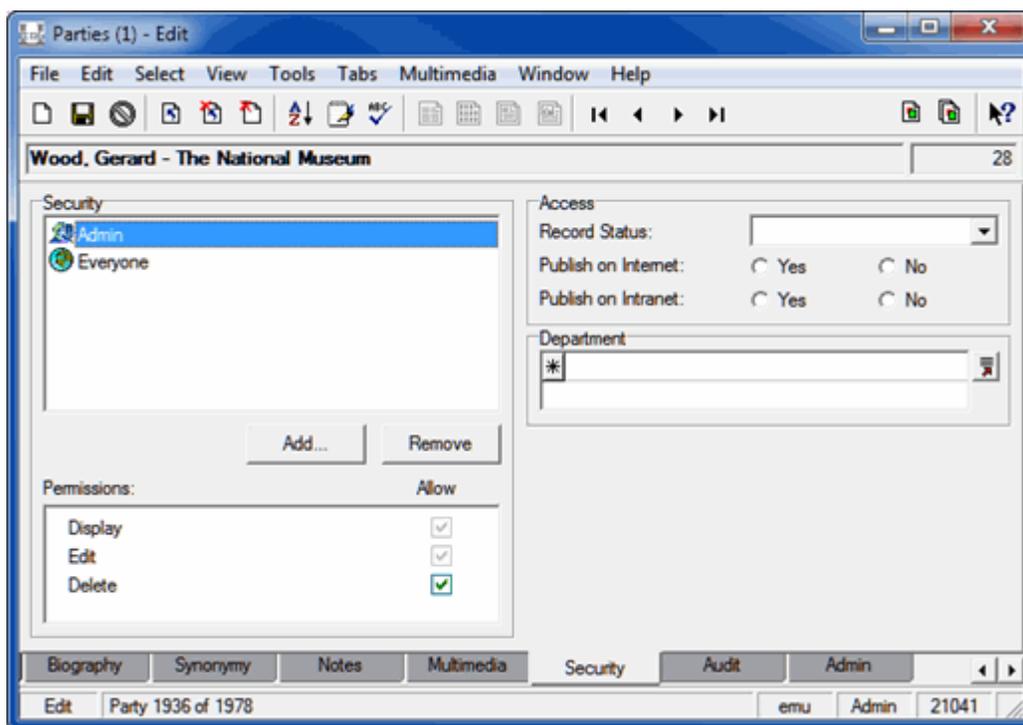
> If your objective is to remove permissions for a user / group and you find that a permission you wish to remove is grayed out and uneditable, a likely reason is that the user / group has inherited the permission from another group to which they belong which has been added to the *Security* box.

How to update permissions for multiple records: batch update

For one record in the previous example (page 6):

1. Permissions for group Everyone were modified by removing the `Edit` and `Delete` permissions.
2. Group Admin was added and given `Edit` and `Delete` permissions.

The Set Record Security tool is designed to apply permissions to multiple records at once:

1. Search for all records for which the same Record Level Security permissions are to apply.
2. For one record, set the required permissions in the *Security* box (add or remove users / groups and tick / untick permissions).
3. Save the record.
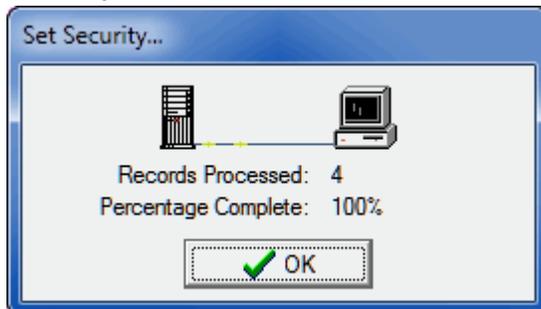4. Select **Tools>Set Record Security>All Records** in the Menu bar.

> If you had selected a subset of records returned by your search at Step 1, you would select **Tools>Set Record Security>Selected Records**.

The following message displays:

5. If you click , the update is processed with progress displayed in the Set Security box:
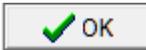
6. Click to close the Set Security box.

Note the following

The Set Record Security tool is designed to batch update values specified in the *Security* box. It is not designed to batch update all values specified on the Security tab. In other words, users / groups added to the *Security* box and permissions specified in the *Permissions* box will be batch updated with the Set Record Security tool, but a value selected in the *Record Status: (Access)* drop list or specified in the *Department* field, for example, will not be batch updated when the Set Record Security tool is run.

> To batch update values in these other fields, use EMu's Global Replace functionality.

Users inherit permissions from groups

For this demonstration of how users inherit permissions from the groups to which they belong, the default group Everyone is given the `Display` permission for a record:
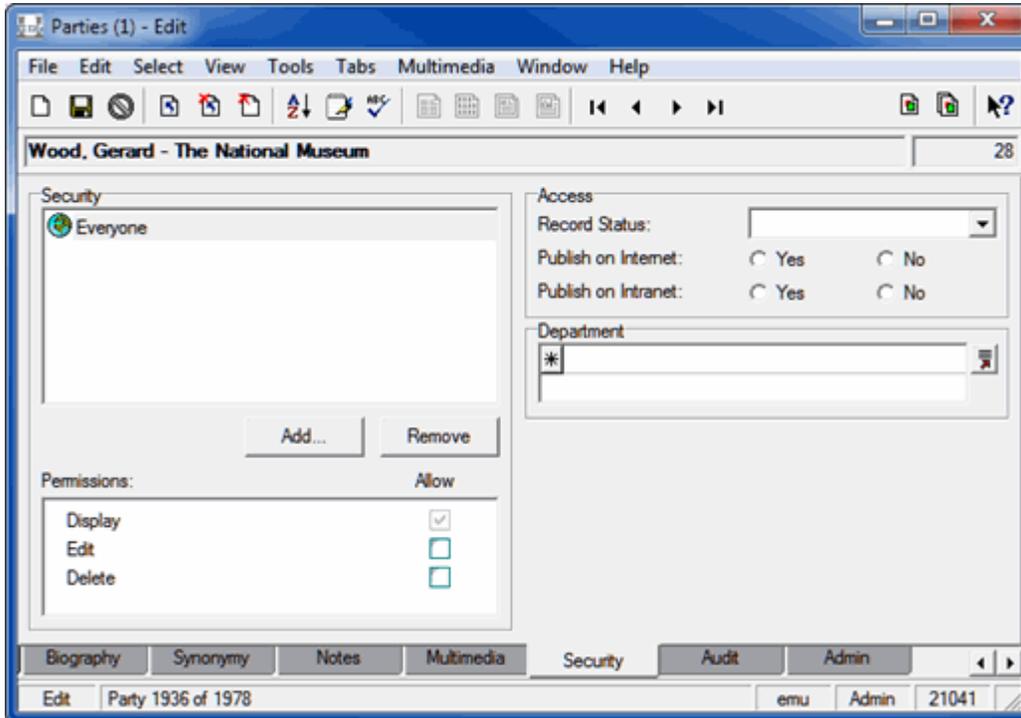
When user gerard is added to the *Security* box he inherits the `Display` permission (which is therefore grayed out and uneditable) as he is a member of group Everyone:
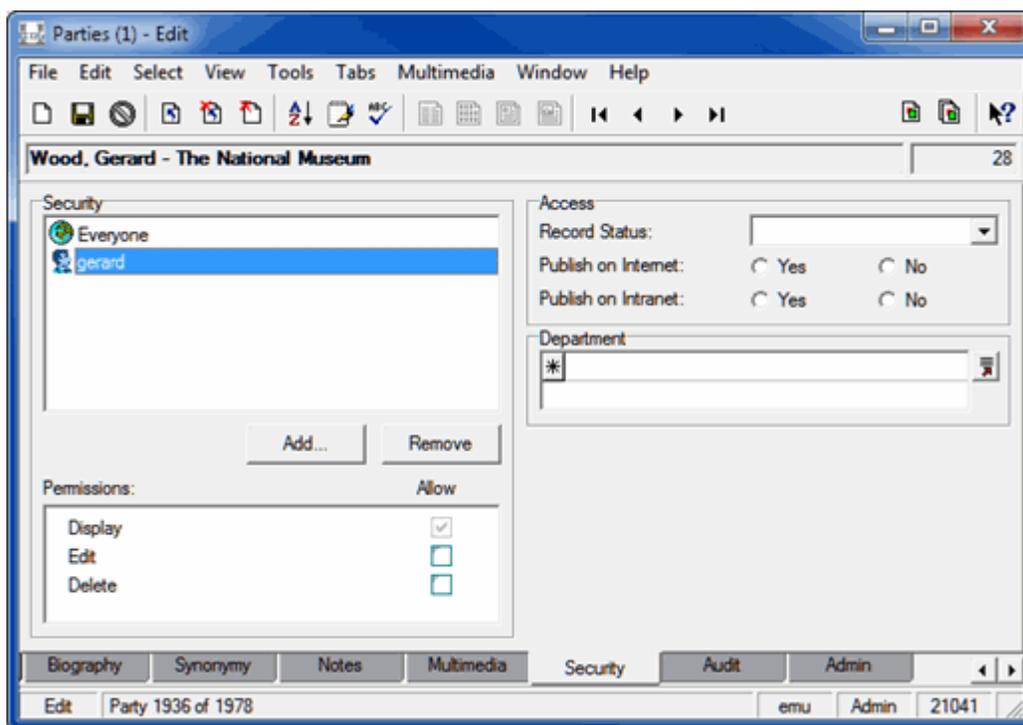
> Technically, the minimum permission a user / group has is the `Display` permission (a user / group added to the *Security* box will always already have the `Display` permission by virtue of being added to the *Security* box).

User gerard can be given both `Edit` and `Delete` permissions (as they are not inherited from group Everyone in this example). In this case we only want to give user gerard the `Edit` permission, but not `Delete`:
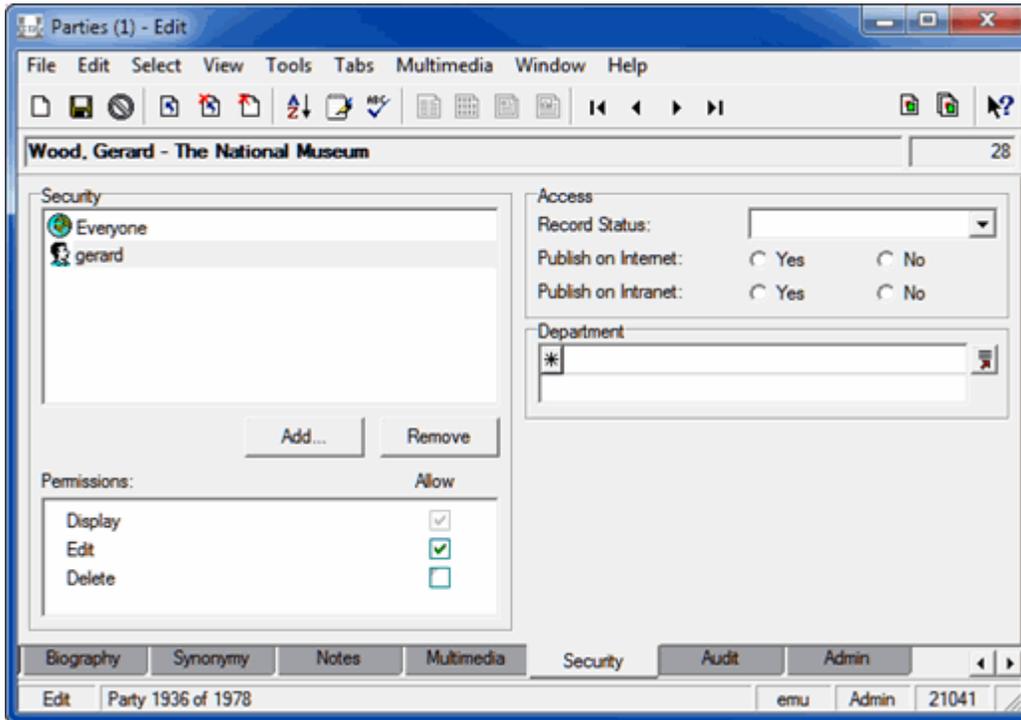
User gerard is also a member of group Curators, which is now added to the *Security* box. This group has both `Edit` and `Delete` permissions:
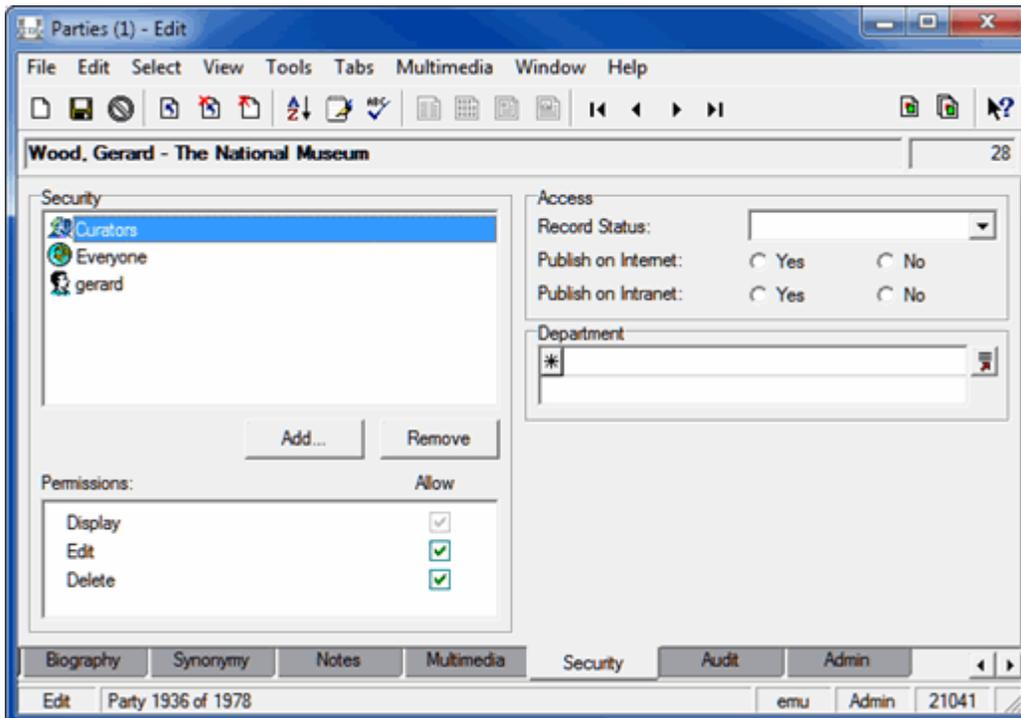
User gerard inherits permissions from all groups to which he belongs when those groups are added to the *Security* box, and now we see that he has the `Delete` permission which we did not want him to have:
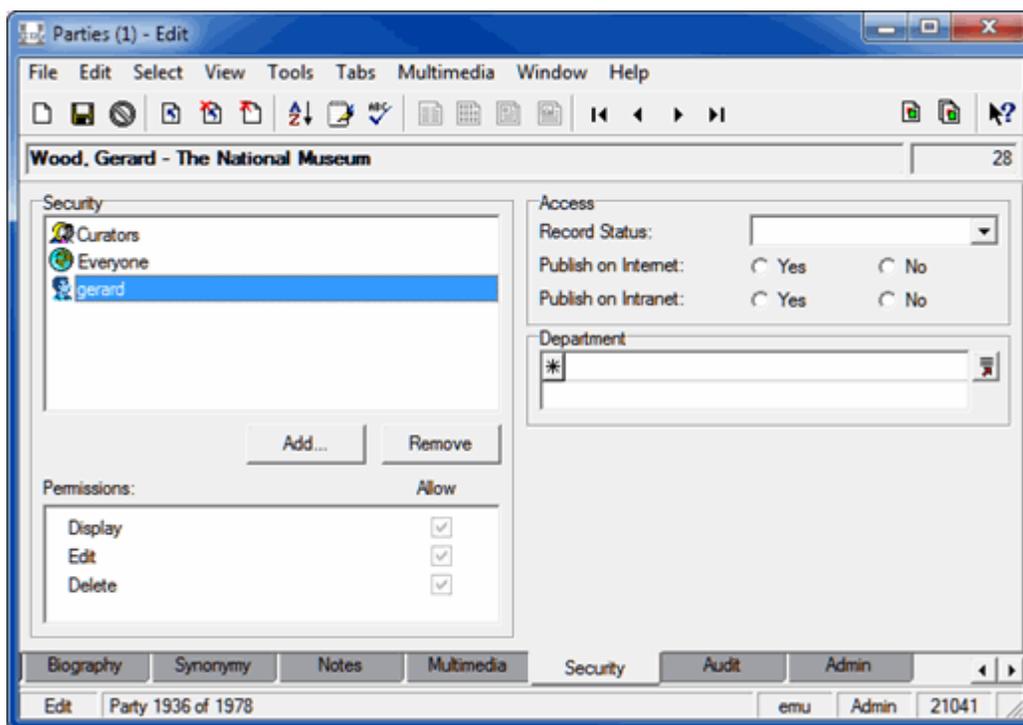
If we only wanted user gerard to have `Edit` permissions but wanted other members of group Curators to have `Edit` and `Delete`, one solution would be to remove user gerard from group Curators.

> If you wish to restrict access to a record, be sure to remove group Everyone from the *Security* box.

How to refine Record Level Security by specifying conditional criteria

At its simplest it is possible to set permissions that determine who can view, edit and delete records. For instance, it is possible to specify that Everyone can view all Parties records but only managers of each department are able to edit and delete the records of staff members in their department. In the following example, group Everyone is able to view the current record (it has the `Display` permission), but only members of group Registrations are able to edit and delete this record (the `Edit` and `Delete` permissions are enabled for group Registrations but disabled for group Everyone):

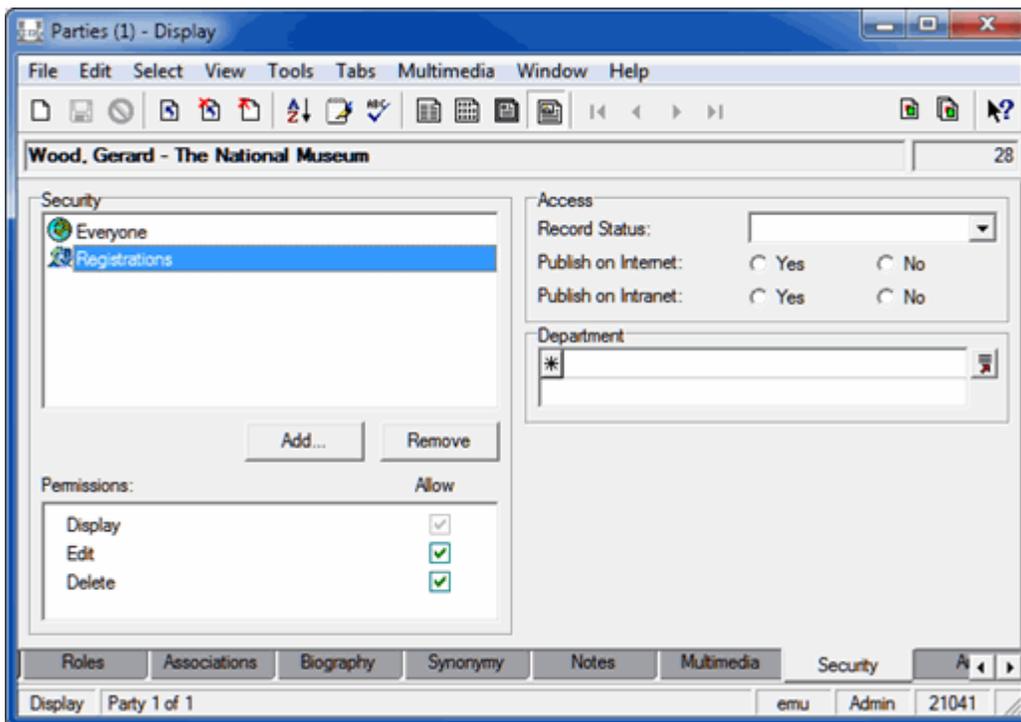With the Security Registry entry (page 21) it is possible to add a level of dynamism to permissions so that the things a user / group can do to a record are conditional upon a value entered in a field (any field) in the module.

In the example above, we manually changed the permissions of group Everyone (limiting it to `Display`) and added the Registrations group to the *Security* box, providing the `Edit` and `Delete` permissions.

With the Security Registry entry it is possible to specify that:

- Members of group Registrations are only able to edit and delete a record if the *Department* field holds the value `Registrations` (in other words, they can only edit and delete their own records)
  -AND-

- When members of group Registrations add a new record:
  - Permissions for group Everyone are limited to `Display`
  - Permissions for group Registrations are set to `Display`, `Edit` and `Delete`
  - The *Department* field is populated with the value `Registrations`

In this way, whenever the value in the *Department* field is updated to hold the value `Registrations` (whether manually or when a new record is added by members of group Registrations), all users will be able to view the record but only members of group Registrations will be able to edit and delete it.

Another useful example of the value of the Security Registry entry is controlling who can view, edit and / or delete a record based on a *Record Status* for instance. If *Record Status* changes from, say, `Active` to `Retired`, permissions can be changed dynamically to hide the record from certain group of users.

> Any field in a module can be used to set conditions when applying Record Level Security. See Security Registry entry (page 21) for details about applying additional security criteria.

We look in some detail at how to limit access to a department's records to members of that department (page 18). The theory is identical when controlling who can access a record based on a value in the *Record Status: (Access)* field, or any other field for that matter.

Example: Restrict Edit and Delete to members of a department

A museum decides that while all curators should be able to view every record in the Catalog module, only curators for each discipline (e.g. Fine Arts, Ceramics, etc.) should be allowed to edit and delete their department's records.

For this example we would need to ensure that:

- Existing records for each discipline are updated with the appropriate values:
    - Permissions are set: `Display` for group Everyone; and `Edit` and `Delete` for members of the group to which the record belongs
      -AND-
    - The Value in the *Department* field is set to the name of the relevant department, e.g. `Fine Arts`, `Ceramics`.

For instance, any Catalog records belonging to the Fine Arts group should have the following permissions and value in the *Department* field:
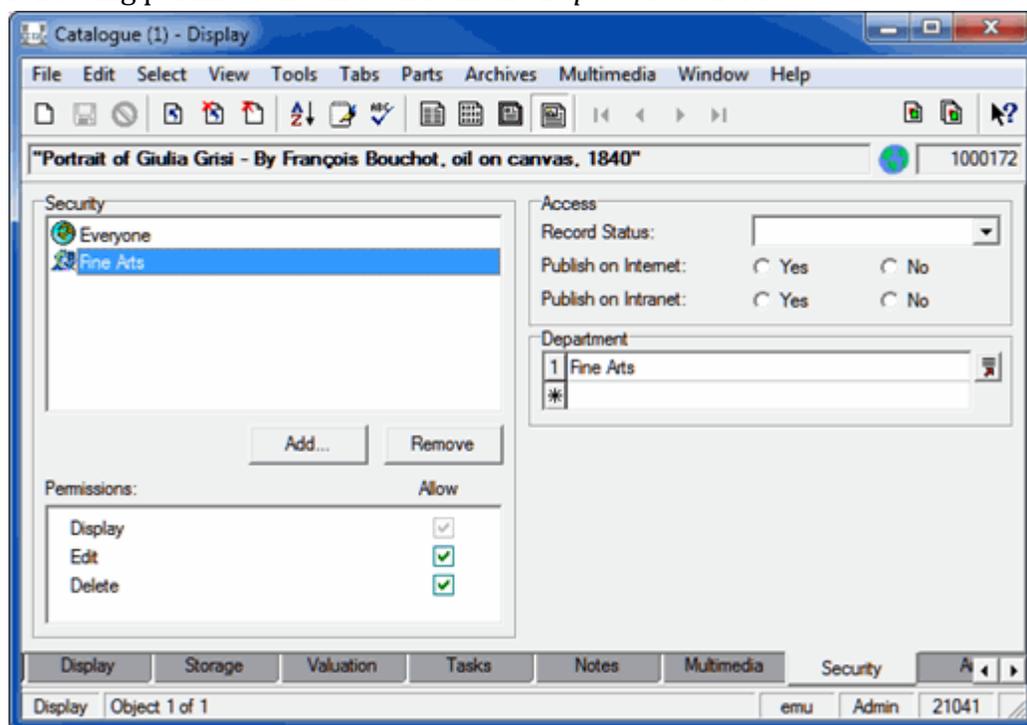
> ℹ️ Using the Set Record Security batch update tool (page 10) it is a simple matter to assign these permissions to existing records.

In order for members of group Fine Arts to edit / delete this record, two Security conditions must be met:

- The Fine Arts group must be added to the *Security* box and must have `Edit` and `Delete` permissions.

> **ℹ** This is necessary because we have removed the `Edit` and `Display` permissions from group Everyone (we don't want everyone to be able to edit or delete this record): if the only group added to the *Security* box is group Everyone, members of group Fine Arts will only inherit the `Display` permission.

- The value in the *Department* field must be `Fine Arts`.

> **ℹ** Although the Set Record Security batch update tool (page 10) cannot be used to batch update the *Department* field, the Global Replace tool can.

- As new records are added by members of a group, the appropriate permissions and values are automatically set.

> **ℹ** See Security Registry entry (page 21) for details of how to refine Record Level Security. The Security Registry entries required for this example can be found here (page 33).

Troubleshoot Record Level Security

Why can't I locate records I know are there? Why, using the same search criteria, are my search results different from those of my colleague?

If a user does not have permission to view (`Display`) a record, it won't be listed in search results.

Why does this Attachment Field display as "Restricted"?

If the Summary Data in an attachment field has been replaced with `Restricted`, the current user does not have permission to view the attached record.

If the user sorts on the attachment field or includes the attachment field in a report, the field will display without any data.

Removing Edit permission to a record

Users can remove their own `Edit` permission to a record, but will not be able to turn it back on again. Only another user with `Edit` permission to that record will be able to reapply the `Edit` permission.

Record Level Security Registry entries

Security Registry entry

Registry Entry	Purpose
`Security`	Used to refine the three standard Record Level Security permissions, `Display`, `Edit`, `Delete`, and to apply the special `Insert` Record Level Security permission.

Overview

Record Level Security provides organizations with control over access to records on a per user and per group basis.

At its simplest it is possible to set permissions that determine who can view, edit and delete records. However, by specifying additional criteria it is possible to refine these three standard security privileges. For instance, it is possible to specify that:

- Records with a *Record Status: (Access)* of `Retired`, cannot be viewed by a certain group of users.

    Any field in a module can be used to set conditions when applying Record Level Security.

- Only members of a department can edit and delete that department's records.

The Security Registry entry is used to refine the three standard Record Level Security permissions, `Display`, `Edit`, `Delete`, and also to apply the special `Insert` Record Level Security permission.

It is worth stressing that a Security Registry *refines* the `Display`, `Edit` and `Delete` permissions that a user / group has to a record, it does not replace the necessity to add the user (or a group to which the user belongs) to the *Security* box on the Security tab for that record and to specify permissions in the *Permissions* box. In other words, a Security entry that gives a user / group the `Edit` permission to a record when *Record Status* is `Active`, is only effective if the user (or a relevant group to which the user belongs) has been added to the *Security* box on the Security tab for that record and the `Edit` checkbox is ticked.

Even if a Security Registry entry is added to allow members of group Curators to edit records in the Parties module when *Record Status: (Access)* = `Active`, members of group Curators will be unable to edit the following record:
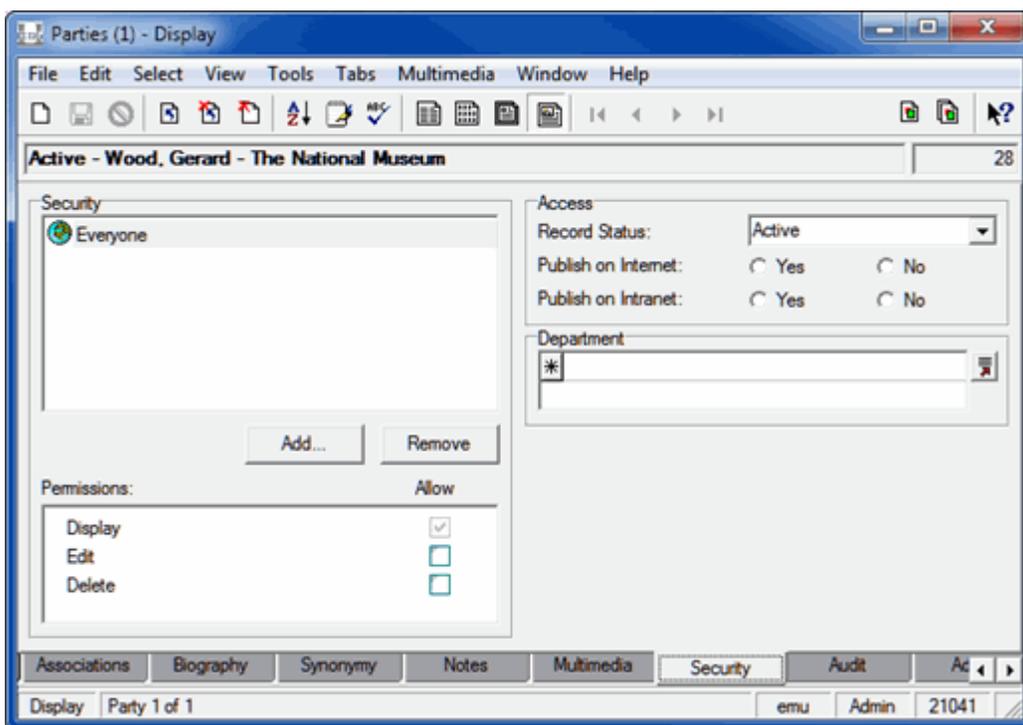
It is necessary to add group Curators to the *Security* box and provide it with the `Edit` permission:
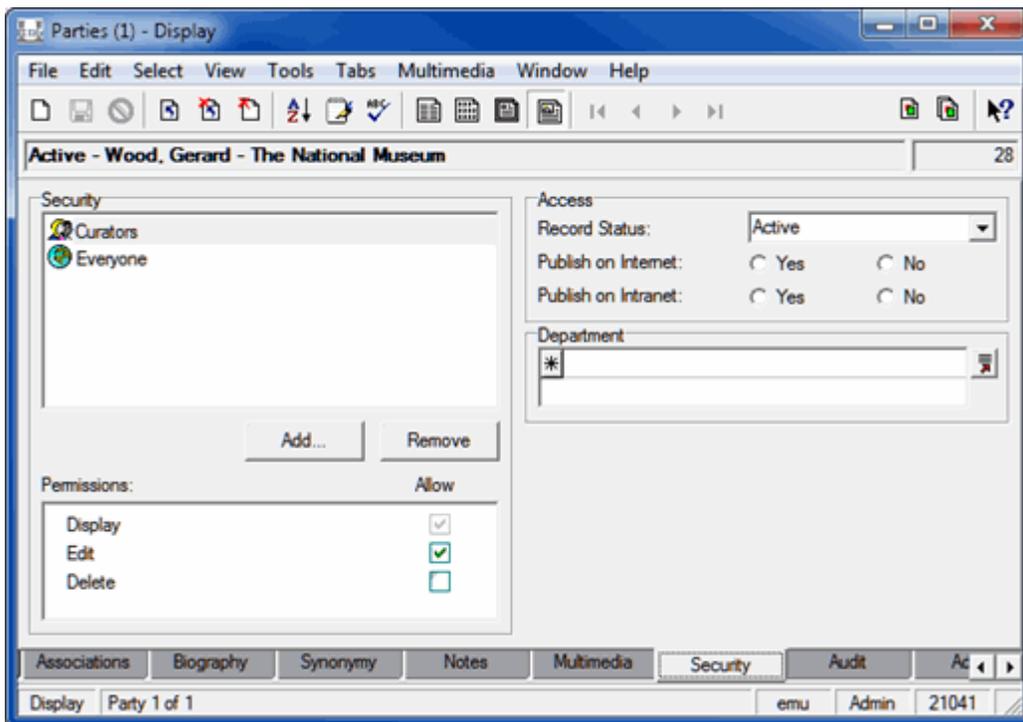
Format of the Registry entry

The format of this Registry entry is:

`User|`*user*`|Table|`*table*`|Security|`*permission*`|`*value;value;...*

`Group|`*group*`|Table|`*table*`|Security|`*permission*`|`*value*

where:

permission	specifies the permission to refine, i.e. `Display`, `Edit` or `Delete`.
	permission can also be `Insert`, a special value that can be used to set permissions for a user / group and to populate a field with a value when a record is added by the user / group. See Example 2 below.
value;value;...	is a semicolon separated list of conditions that must be met for the *permission* to apply. This is in the format: *column=value*
	e.g. *SecRecordStatus=Active*
	It is also possible to embed the user / group name of the currently logged in user (stored by EMu as `$user` and `$group`) in security values. Using these variables, security may be adjusted on a per user / group basis depending on one or more user / group names stored in the data. See Example 2 below.

> When referencing an attachment field in a Security Registry entry, it is necessary to use a field's *Link Column* name and a record's IRN. See Example 3 below.

Example 1

This Registry entry allows members of group Curators to view records in the Parties module only when *Record Status: (Access)* = `Active`:

Field	Value	Description
Key 1	`Group`	Group or User
Key 2	`Curators`	Group or User name
Key 3	`Table`	Table
Key 4	`eparties`	Name of the table for which permissions are to be set
Key 5	`Security`	
Key 6	`Display`	Type of permission: `Display`, `Edit`, `Delete`, `Insert`.
Value	`SecRecordStatus=Active`	The condition that must be met for the permission to apply. In this case, if *Record Status* is set to `Active`, members of group Curators will be able to view the record. If *Record Status* is anything other than `Active`, members of group Curators will be unable to view the record. Any field can be used to set a security permission. Multiple values (multiple conditions) can be set here.

In this example:

- Group Everyone can display, edit and delete the following record in the Parties module.
- *Record Status: (Access)* is set to `Active`:
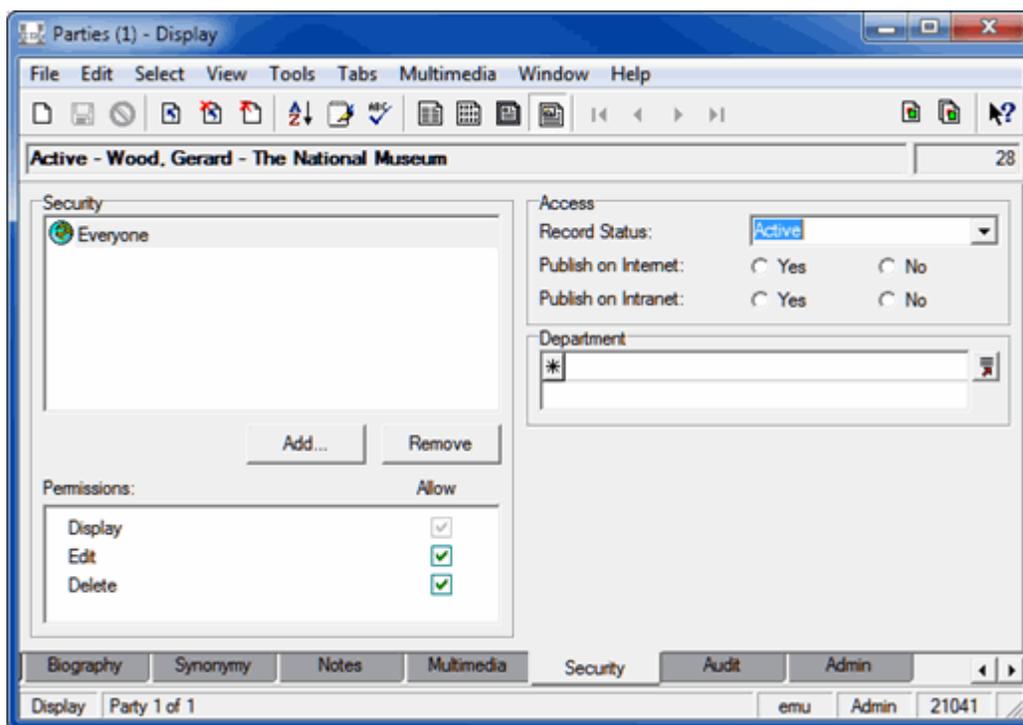
When a member of group Curators logs in and searches for this record, the record is returned and the user has inherited the permissions of group Everyone (they can edit and delete the record).

If *Record Status: (Access)* is changed from `Active` to `Retired`, and a member of group Curators searches for this record, the record will not be located:
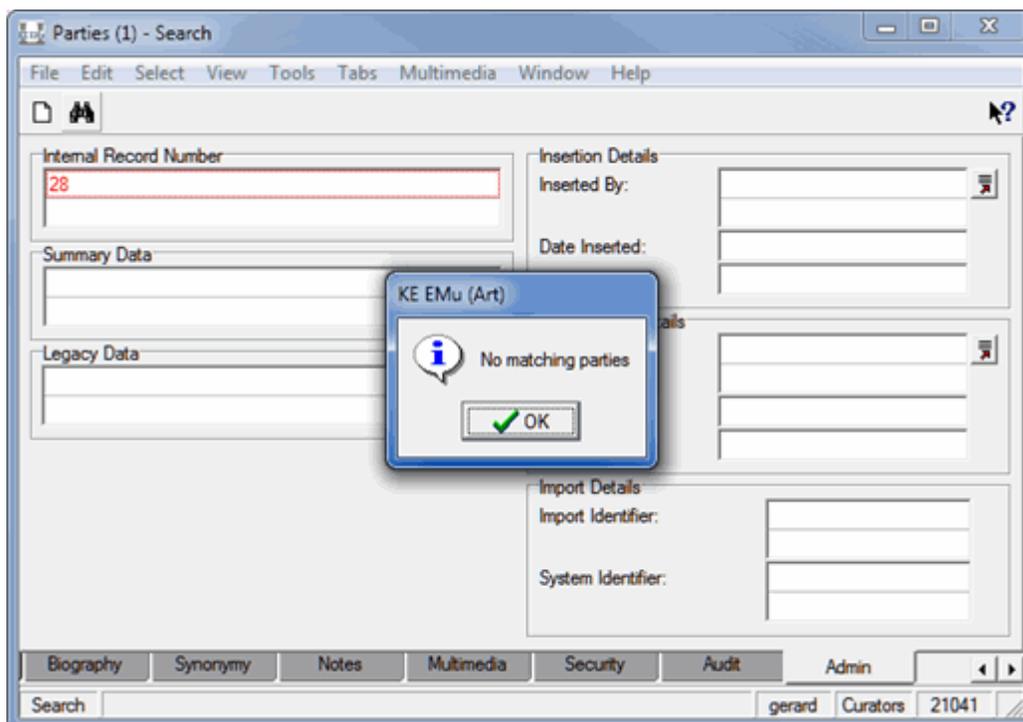
Even if group Curators is added to the *Security* box for this record:
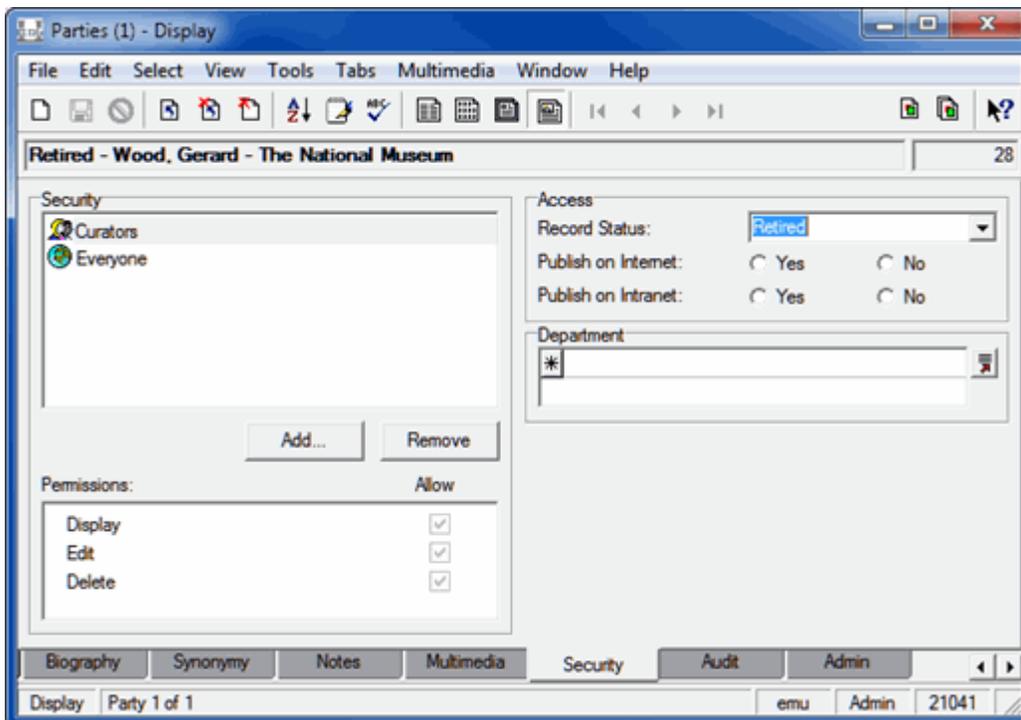
members of group Curators will not be able to display the record unless *Record Status: (Access)* is set to `Active`.

Note that it is not necessary to add group Curators to the *Security* box as long as group Everyone *is* listed in the *Security* box: members of group Curators inherit the permissions of any other group to which they belong, which by default includes group Everyone. If the above Security Registry has been set and a record has the following security values, members of group Curators will be able to view, edit and delete the record even though group Curators has not been added to the *Security* box:
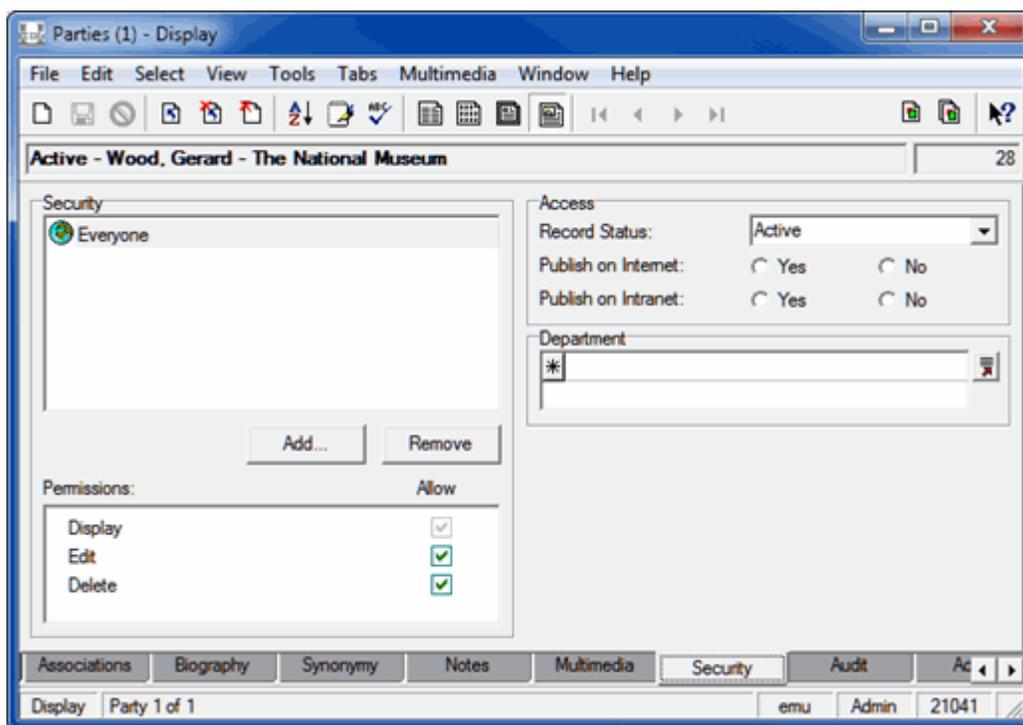
However, if we take away the `Edit` and `Delete` permissions from group Everyone:
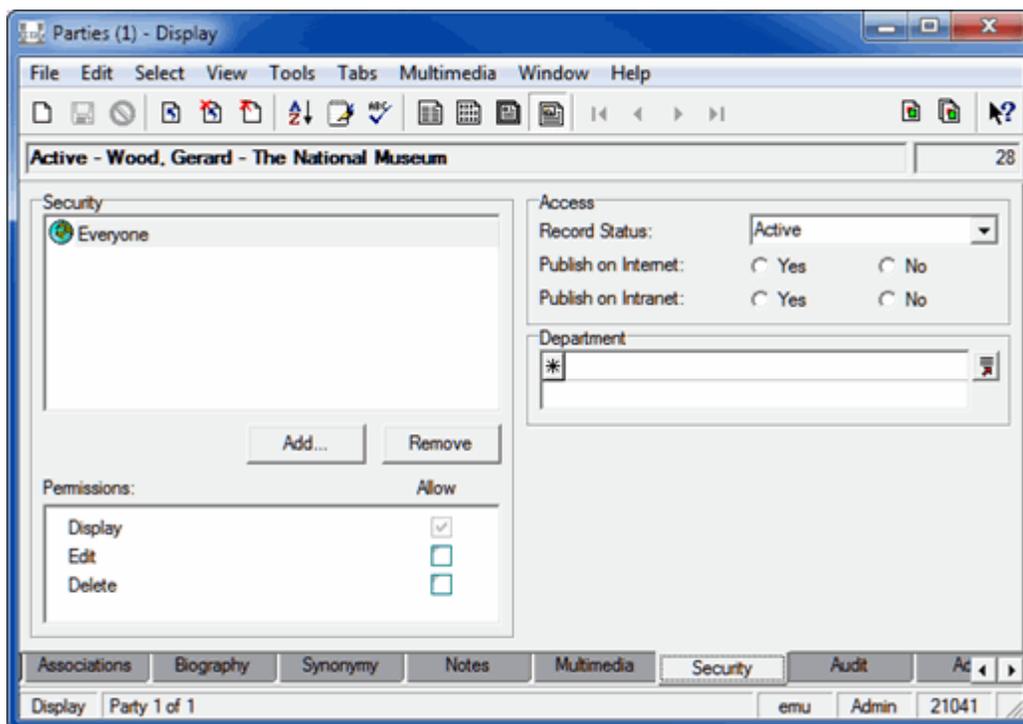
and wish to provide group Curators with the ability to edit records where *Record Status: (Access)* = `Active`, then it is necessary to add group Curators to the *Security* box and provide it with the `Edit` permission:
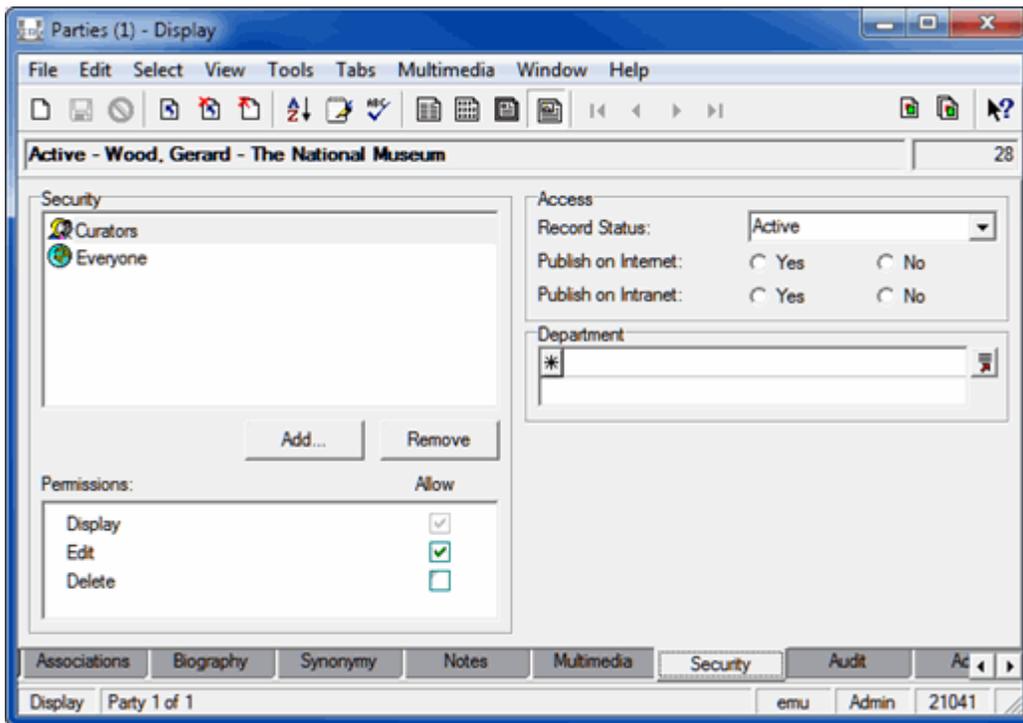
Example 2

When members of group Curators add a new record to the Parties module, the *Department* field on the Security tab is populated with `Curators`:

Field	Value
Key 1	`Group`
Key 2	`Curators`
Key 3	`Table`
Key 4	`eparties`
Key 5	`Security`
Key 6	`Insert`
Value	`SecDepartment_tab=Curators`
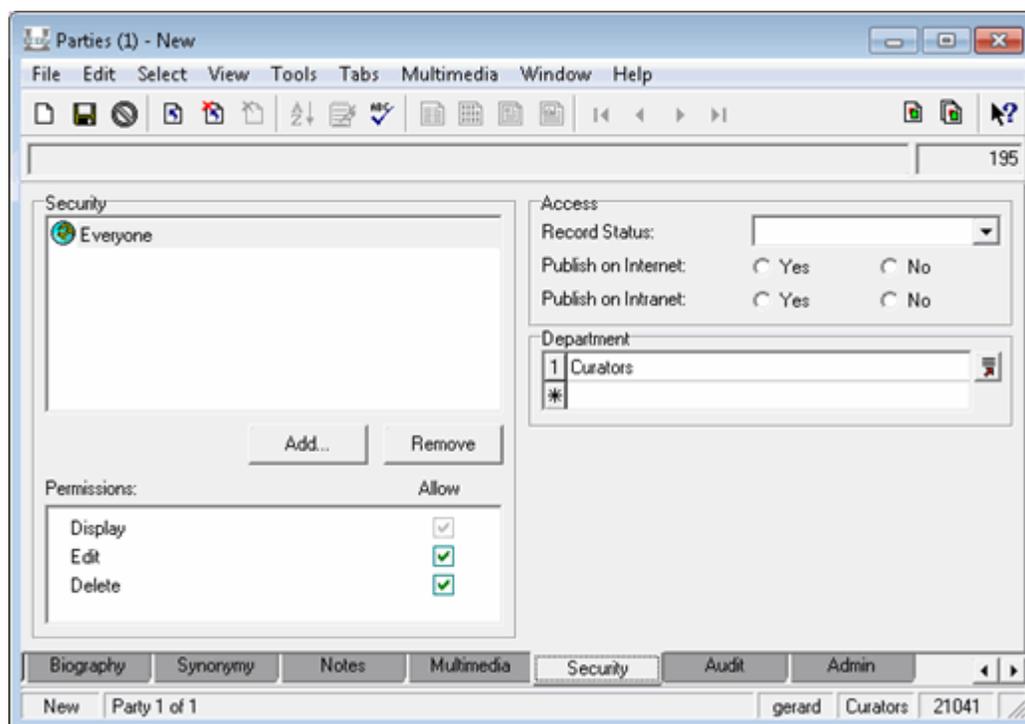
The following entry not only inserts a value in the *Department* field when group `Curators` adds a record to the Parties module, it also specifies the security permissions for group Everyone and the current group (`Curators`) using the `$group` variable:

Field	Value
Key 1	`Group`
Key 2	`Curators`
Key 3	`Table`
Key 4	`eparties`
Key 5	`Security`
Key 6	`Insert`
Value	`SecDepartment_tab=Curators; SecCanDisplay=Group Default; SecCanDisplay=Group $group; SecCanEdit=Group $group; SecCanDelete=Group $group`

Example 3

When specifying an attachment field in a Security Registry entry, the *value* must be:

1. The field's *Link Column* name, not its *Column* name
2. And a record's IRN.

For example, if referencing the *Party: (Associated With)* field in a Security Registry entry, we would use *AssAssociationRef_tab* and not *AssAssociation_tab*:

A suitable Security Registry entry referencing an attachment field would be:

Field	Value
Key 1	Group
Key 2	Curators
Key 3	Table
Key 4	eparties
Key 5	Security
Key 6	Display
Value	AssAssociationRef_tab=6

Example

A museum decides that while all curators should be able to view every record in the Catalog module, only curators for each discipline (e.g. Fine Arts, Ceramics, etc.) should be allowed to edit and delete their department's records.

In this example we restrict records to members of the following groups:

- Fine Arts Curators
- Ceramics Curators

Suitable Registry entries are:

1. `Group|Fine     Arts     Curators|Table|ecatalogue|Security|Edit| SecDepartment_tab=Fine Arts`

    This entry specifies that members of group Fine Arts Curators are able to edit records in the Catalog module when the *Department* field on the Security tab holds the value `Fine Arts`.

2. `Group|Fine                                           Arts Curators|Table|ecatalogue|Security|Delete|SecDepartment_tab= Fine Arts`

    This entry specifies that members of group Fine Arts Curators are able to delete records in the Catalog module when the *Department* field on the Security tab holds the value `Fine Arts`.

3. `Group|Fine                                           Arts Curators|Table|ecatalogue|Security|Insert|SecDepartment_tab= Fine    Arts;SecCanDisplay=Group    Default;SecCanDisplay=Group $group;SecCanEdit=Group $group;SecCanDelete=Group $group`

    This entry specifies that when members of group Fine Arts Curators add a record to the Catalog module, the *Department* field on the Security tab is populated with the value `Fine Arts` and permissions are set which allow everyone to view the record but only members of group Fine Arts Curators to edit and delete the record.

4. `Group|Ceramics Curators|Table|ecatalogue|Security|Edit|SecDepartment_tab= Ceramics`

    This entry specifies that members of group Ceramics Curators are able to edit records in the Catalog module when the *Department* field on the Security tab holds the value `Ceramics`.

5. `Group|Ceramics Curators|Table|ecatalogue|Security|Delete|SecDepartment_tab= Ceramics`

    This entry specifies that members of group Ceramics Curators are able to delete records in the Catalog module when the *Department* field on the Security tab holds the value `Ceramics`.

6.  ```
    Group|Ceramics        Curators|Table|ecatalogue|Security|Insert|
    SecDepartment_tab=Ceramics;SecCanDisplay=Group
    Default;SecCanDisplay=Group              $group;SecCanEdit=Group
    $group;SecCanDelete=Group $group
    ```

    This entry specifies that when members of group Ceramics Curators add a record to the Catalog module, the *Department* field on the Security tab is populated with the value `Ceramics` and permissions are set which allow everyone to view the record but only members of the Ceramics Curators group to edit and delete the records.

When these entries have been set in the Registry module, all that remains is to apply these settings to existing records in the Catalog module:

1. In the Catalog module, locate all Fine Arts records.
2. Add group Fine Arts Curators to the *Security* box on the Security tab and give the group `Edit` and `Delete` permissions.
3. Uncheck all but the `Display` permission for the Everyone group.
4. Use the Set Record Security batch update tool (page 10) to apply these settings to all Fine Arts records.
5. Use the Global Replace tool to insert the value **Fine Arts** in the *Department* field on the Security tab for all records found in your search.

    Now when Curators in the Fine Arts Department log in, they will have edit and delete permissions for these Fine Arts records; other users will be able to view them but not edit or delete them.
6. Repeat steps 1 to 5 but this time for group Ceramics.

Disabling Record Level Security

Record Level Security can be disabled on an individual table basis by creating a file called *emuoptions* in the table's database directory on the EMu server, and adding the line:

```
SECURITY=no
```

Once added, running *emusecurity* will disable security on the database. Running *emusecurity* generates new security profiles for EMu.

To run the utility, in the Registry module:

1. Select **Tools>Generate Record Security** in the Menu bar.

> To be able to run the *emusecurity* utility a user must have (or be a member of a group that has) the *daSecurity* Operations permission. Typically this will be set at the System level for the Admin group.

> The Security tab may still be visible within the EMu client. If so, the tab should be hidden via suitable tab switching Registry entries.

Generate Record Security: emusecurity

> As of EMu 4.0.03, security profiles are generated automatically after saving any user registration based Registry entries and it is no longer necessary to run **Tools>Generate Record Security** after changing user settings.

The *emusecurity* utility must be run:

1. Whenever a user / group is added to the Registry.
2. Whenever a user changes groups.
3. When a user is removed from EMu.
4. Whenever the Security Registry entry is modified.

Running *emusecurity* generates new security profiles for EMu. If it is not run after these changes to the System, users will not have the appropriate Record Level Security settings when they log in to EMu.

To run the utility, in the Registry module:

1. Select **Tools>Generate Record Security** in the Menu bar.

> To be able to run the *emusecurity* utility a user must have (or be a member of a group that has) the *daSecurity* Operations permission. Typically this will be set at the System level for the Admin group.

Dynamic Security

The default security model used by EMu is static in nature. User and Group privileges are defined in the Registry and loaded into a module when it is invoked. Once invoked the security of the module remains the same throughout its lifetime. If a user can change the contents of a given field, then they can change it for all records (assuming Record Level Security (page 21) allows the record to be modified). In some instances it would be useful to allow some security settings to be altered based on information stored in the current record.

For example:

1. Storage staff in a group called Storage are able to change all fields for records of type `Crate` and `Frame`, while for `Object` records they can only update the dimensions information. The current EMu security model does not provide a mechanism for implementing this requirement via Registry settings. It is possible to "hard wire" such functionality into the EMu client, however it becomes very difficult to change as new requirements arise.

   What would be useful is a mechanism that allows access to a column to be modified based on the contents of the record.

2. Similarly, you may require certain fields to be filled depending on the type of record. For `Object` records you may require the *Main Title* field to be specified, while for `Crate` and `Frame` records the title should not be specified (in fact, it should not even be shown). The EMu Mandatory Registry entry allows a field to be defined as mandatory, however it is not possible to use this Registry entry to specify conditional mandatory settings.

   As with the first example, it would be useful to allow the mandatory setting for a field to be set based on the contents of the record.

3. Alternatively, you may want to alter Record Level Security settings based on the contents of data within the record when the record is saved. One such requirement may be that records whose record status is set to `Retired` may only be edited by users in group Admin. Such a feature can be "hard wired" into the database server.

   However a solution that uses the Registry would provide a more flexible mechanism.

EMu 4.1 onwards has three Registry entries which provide for a flexible and dynamic security model that can adapt based on the data stored within a record:

- The Column Access Modifier Registry entry handles the first example above, that is the ability to alter column access based on the contents of the record.
- The Mandatory Modifier Registry entry handles the second example above, that is the ability to adjust mandatory settings based on the contents of the record.
- The Security|Update Registry entry (page 39) provides for the third example, that

is the ability to change Record Level Security based on the contents of the current record.

> Click the links above for full details of each entry.

The combination of the these facilities provides a useful mechanism for altering the EMu security settings dynamically. The use of dynamic security allows very flexible security models to be implemented.

Security|Update Registry entry

Registry Entry	Purpose
Security|Update	Used to update the contents of one or more fields in the current record when the record is saved if a condition is met (a given value is in a field in the current record). Notably (but not exclusively) used to modify Record Level Security security fields.

Overview

This functionality is available in EMu 4.1 or later. Texpress 8.3.006 (or later) is required.

The Security Update Registry entry allows the contents of one or more fields to be modified whenever a record is saved (inserted or modified) conditional on the value in a field in the current record. Importantly, the Record Level Security (RLS) fields:

- SecCanDisplay
- SecCanEdit
- SecCanDelete

may be updated, allowing the record security to be modified if a condition in the current record changes.

The Security|Update Registry entry is not limited to the update of RLS fields however, and any fields may be adjusted on record save based on the value in a field in the current record.

Format of the Registry entry

The format of this Registry entry is:

```
User|user|Table|table|Security|Update|column|value|settings
User|user|Table|Default|Security|Update|column|value|settings
Group|group|Table|table|Security|Update|column|value|settings
Group|group|Table|Default|Security|Update|column|value|settings
Group|Default|Table|table|Security|Update|column|value|settings
Group|Default|Table|Default|Security|Update|column|value|settings
```

where:

column defines which field should be consulted to look for a matching *value*.

value	is an EMu based search pattern.

If there is a match of the data in *column* with the *value* query, then the Registry entry is used and *settings* is applied.

> If *column* contains a table of values, each entry is checked against *value*.

Since *value* is a pattern, particular attention must be paid if you want to match the complete contents of a field.

For example, to match `Registration` but not `Pending Registration`, the pattern `^Registration$` should be used. It is important to remember that *value* operates in the same way as an EMu search term.

All comparisons of *value* are case insensitive.

settings	is a semicolon separated list of assignments to columns that is applied if there is a match of the data in *column* with the *value* query.

The format of *settings* is:

column`=[+/-]`*term*`:[+/-]`*term*`:...;`*column*`=[+/-]`*term*`:...`

where:

column	is the name of the column to be modified.
term	is the value to be set in *column.*

- If *term* is preceded by a plus symbol (+), the value is added to the existing list of values.
- If *term* is preceded by a minus sign (–), the value is removed from the existing list of values.
- If no symbol precedes a *term*, the current contents of *column* are removed and replaced with *term*.
- If more than one *term* is supplied for a given column, separated by colons (`:`), each *term* is applied one after the other.
- If more than one column is to be modified, each set of column settings is separated by a semicolon (`;`).

See Security Registry entry (page 21) for more details.

How to reference an attachment field in a Security|Update Registry entry

When referencing an attachment field in a Security|Update Registry entry, it is necessary to use a field's *Link Column* name and a record's IRN. This applies whether referencing the attachment field as *Key 7* (*column*) or as the Registry entry *settings.* For example:

Field	Value	
Key 1	Group	
Key 2	Admin	
Key 3	Table	
Key 4	ecatalogue	
Key 5	Security	
Key 6	Update	
Key 7	LocCurrentLocationRef	The attachment field is referenced as *Key 7*.
Key 8	9800	A record's IRN is required at *Key 8*.
Value	SecRecordStatus=Deaccession	

Field	Value	
Key 1	Group	
Key 2	Admin	
Key 3	Table	
Key 4	ecatalogue	
Key 5	Security	
Key 6	Update	
Key 7	SecRecordStatus	
Key 8	^Deaccession$	
Value	LocCurrentLocationRef=9800	When an attachment field is referenced as the value in a Security|Update Registry entry, it is in the format `Link Column name=IRN`.

Examples

If we want Record Level Security to be adjusted so that users in group Admin are the only ones allowed to edit and delete records that have a record status of `Retired`, the following entry can be used:

Field	Value
Key 1	Group
Key 2	Default
Key 3	Table
Key 4	Default
Key 5	Security
Key 6	Update
Key 7	SecRecordStatus
Key 8	^Retired$
Value	SecCanEdit=Group Admin;SecCanDelete=Group Admin

Keys *7* and *8* indicate that the entry only applies where the *SecRecordStatus* column matches the pattern `^Retired$` (i.e. where the field contains `Retired` only). If this is not the case, then the Registry entry is ignored. Where a record does match, the *SecCanEdit* field is set to group Admin. As a leading plus or minus is not supplied, the contents of *SecCanEdit* are replaced with group Admin. A similar update occurs for *SecCanDelete*.

A more complex example would involve removing edit access for all users in groups Conservation and Storage when an object is deaccessioned. Let's assume that the column *RecObjectStatus* contains the word `Deaccessioned` for objects that are no longer part of our collection. A suitable Registry entry would be:

Field	Value
Key 1	Group
Key 2	Default
Key 3	Table
Key 4	ecatalogue
Key 5	Security
Key 6	Update
Key 7	RecObjectStatus
Key 8	^Deaccessioned$
Value	SecCanEdit=-Group Conservation:-Group Storage

Notice how the terms to set have a leading minus sign, indicating the term (in this case the group name) is to be removed from the field *SecCanEdit*.

A third example requires all records with an object valuation of `High` to have group Student removed and group Valuers added for both displaying and editing the record. The field containing the object valuation is *ValValuationCode*. A suitable Registry entry is:

Field	Value
Key 1	`Group`
Key 2	`Default`
Key 3	`Table`
Key 4	`ecatalogue`
Key 5	`Security`
Key 6	`Update`
Key 7	`ValValuationCode`
Key 8	`^High$`
Value	`SecCanDisplay=-Group Student:+Group Valuers; SecCanEdit=-Group Student:+Group Valuers`

Notice how more than one field may be updated with a single Registry entry. Note also:

- If a term has a leading plus symbol and the term already appears in the field, it is not added again.
- Similarly, if a term has a preceding minus and it does not appear in the field, it is ignored.

In this final example we restrict the display privilege for records that have not been approved for viewing on the intranet to groups Admin, Curator, Storage and Conservation. Once the record has been approved for viewing on the intranet we will allow all users to view the record. In this case two Registry entries are required:

- The first restricts access for records not available on the intranet.
- The second allows access to all users for record available on the intranet.

Suitable Registry entries are:

Field	Value
Key 1	Group
Key 2	Default
Key 3	Table
Key 4	ecatalogue
Key 5	Security
Key 6	Update
Key 7	AdmPublishWebPasswordFlag
Key 8	N
Value	SecCanDisplay=Group Admin:+Group Curator:+Group Storage:+Group Conservation

Field	Value
Key 1	Group
Key 2	Default
Key 3	Table
Key 4	ecatalogue
Key 5	Security
Key 6	Update
Key 7	AdmPublishWebPasswordFlag
Key 8	Y
Value	SecCanDisplay=Group Default

Notice how the first term in the first Registry entry (`Group Admin`) does not have a leading plus or minus, meaning it replaces the current contents of *SecCanDisplay*. The following terms require a leading plus symbol otherwise they will also clear the current contents rather than adding to the first term. The second Registry entry replaces the contents of *SecCanDisplay* with group Default (this allows access for everyone).

By using a combination of Registry entries it is possible to produce quite sophisticated and dynamic privilege changes.

What's happening behind the scenes

The Security Update Registry entry is a Record Level Security based Registry entry and like all RLS Registry entries it is enforced by the database engine. The `security` file stored in the table directory provides the configuration used for RLS. The XML format of the `security` file has been extended to allow the contents of the Security|Update Registry entry to be accommodated. Consider this entry:

Field	Value
Key 1	`Group`
Key 2	`Default`
Key 3	`Table`
Key 4	`ecatalogue`
Key 5	`Security`
Key 6	`Update`
Key 7	`SecRecordStatus`
Key 8	`^Retired$`
Value	`SecCanEdit=Group Admin:+Group Registration;SecCanDelete=Group Admin:+Group Registration`

The extra XML generated in the `security` file for this entry is:

```
<updates>
  <update name="SecRecordStatus" value="^Retired$">
    <columns>
      <column name="SecCanEdit">
        <values>
          <value operation="replace" term="Group Admin"/>
          <value operation="add" term="Group Registration"/>
        </values>
      </column>
      <column name="SecCanDelete">
        <values>
          <value operation="replace" term="Group Admin"/>
          <value operation="add" term="Group Registration"/>
        </values>
      </column>
    </columns>
  </update>
</updates>
```

As you can see the XML follows the sequencing of the Registry entry. If multiple Registry entries exist, the `<update>` tags are repeated.

The good news is that you do not need to add the XML to the `security` file. Whenever a security based Registry is added or modified in EMu, the `security` file is rebuilt automatically, and all you need to do is add the Registry entries.

The order of processing

The database server applies the Security|Update settings whenever a record is saved (i.e. for all insertions and updates). The entries are applied after assignment expressions have been executed and **before** validation is run. This means that assignment expressions may be used to build a composite value that may be tested by Security|Update settings. For example, it is possible to concatenate two fields into one that may then be checked.

It is also possible to apply sophisticated formula to calculate a field to be checked. For example, you may want to set Security|Update Registry entries based on a range of valuations for an object. You could use an assignment expression to set a value in a field based on the ranges:

Range	Value
$0 - $1000	`Cheap`
$1001 - $10000	`Average`
$10001 -	`Pricey`

You may then use the three values, `Cheap`, `Average` and `Pricey` in Security|Update Registry entries. Note that you cannot use validation code to compute values as the Security|Update entries are applied before validation code is executed.

Index