7 Ways to Lock a Record in Salesforce

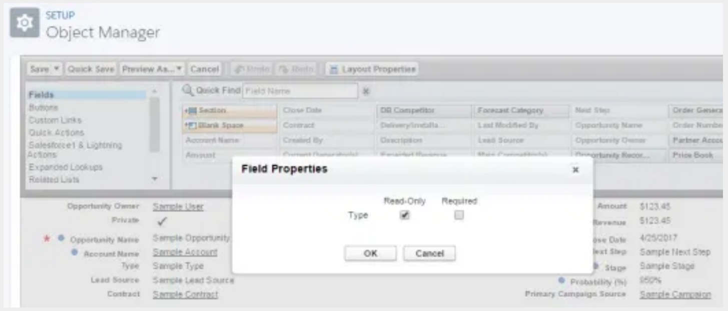
In this blog post, I'd be discussing the different ways to **lock** a Salesforce record in UI. By lock, I mean, the user shouldn't be able to edit the field values of the record. I have identified 7 ways to lock a record using configuration and coding.

There is a general requirement that the Business doesn't want any user to edit any fields of an Opportunity record when it is in 'Finalization' stage or 'Deal Closing' stage or even when the opportunity is 'Closed Won'. Not just opportunity, it could be any object (standard/custom) sometimes the business doesn't want user to edit a record when it is in a certain stage of the sales/business process.

There is no "best" approach among these 7, each approach has its own pros and cons. The approach you choose purely depends on what suits your requirement. I will highlight the pros and cons for each of these approaches, so that it helps you to make a smarter decision.

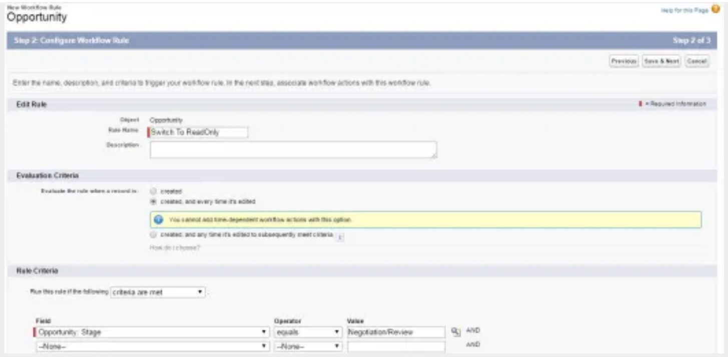
1. Record Types & Workflow/Process Builder

In this approach, you need to create 2 page layouts. I'm calling them – "EditablePageLayout" and "ReadOnlyPageLayout". The editable page layout is the existing normal page layout you already have. The read only page layout will have all the fields in read-only mode i.e., for every field in this page layout, you have to click the settings icon and check the 'Read Only' check box as shown below.



Once you have the page layouts ready, create 2 record types and assign the page layouts to the record types.

Now create a workflow rule – when opportunity stage is in "Negotiation/Review", we need to change the page layout from EditablePageLayout to ReadOnlyLayout. To change the pagelayout, we need to change the record type for that record using workflow. Here's how the workflow looks like



New Workflow Rule
Opportunity

Step 2: Configure Workflow Rule — Step 2 of 3

Enter the name, description, and criteria to trigger your workflow rule. In the next step, associate workflow actions with this workflow rule.

Edit Rule

Object	Opportunity
Rule Name	Switch To ReadOnly
Description	

Evaluation Criteria

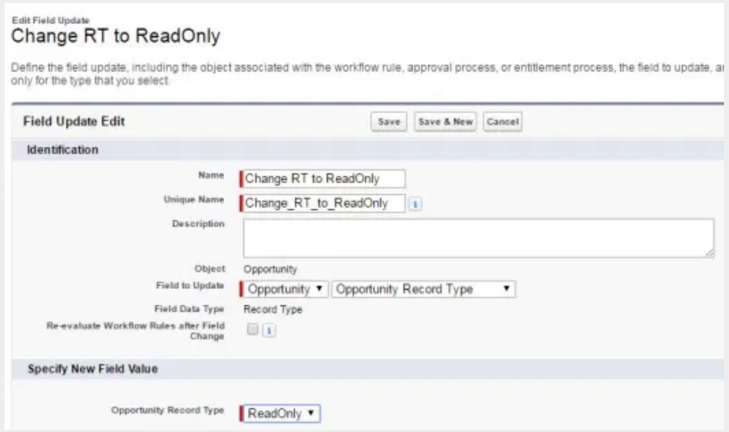
Evaluate the rule when a record is:
- created
- [✓ created, and every time it's edited]
- You cannot add time-dependent workflow actions with this option.
- created, and any time it's edited to subsequently meet criteria

Rule Criteria

Run this rule if the following: criteria are met

Field	Operator	Value	
Opportunity: Stage	equals	Negotiation/Review	AND
--None--	--None--		AND

And then create a workflow field update to set the Record type.



Edit Field Update
Change RT to ReadOnly

Define the field update, including the object associated with the workflow rule, approval process, or entitlement process, the field to update, a only for the type that you select.

Field Update Edit

Identification

Name	Change RT to ReadOnly
Unique Name	Change_RT_to_ReadOnly
Description	
Object	Opportunity
Field to Update	Opportunity / Opportunity Record Type
Field Data Type	Record Type
Re-evaluate Workflow Rules after Field Change	☐

Specify New Field Value

Opportunity Record Type	ReadOnly

That's it, save the workflow field update and activate the workflow. When the opportunity is moved to 'Negotiation/Review' stage, the page layout changes to ReadOnlyPagelayout and the users cannot edit it anymore.

Note: With the 1st approach, every time user creates a new Opportunity, the user is asked to choose the record type! To skip this, you just have to remove the ReadOnly Record Type from the user's Profile (and set the main record type as default). Even without record type permission, the user will still be able to view the Readonly record. Also, System Administrator would be still able to edit the record.

Tip : Instead of a workflow, you can of course use Process Builder to implement the logic for switching of record types from Editable to Read Only.

2. Record Types & Trigger

This approach is exactly same as the 1st one except that instead of workflow, we use a Trigger. You have to create 2 page layouts, 2 record types as mentioned in the 1st one. Except for the workflow, the change of RecordType logic is implemented in an Apex trigger.

Here’s the trigger code to change RecordType for the opportunity.

```
trigger OppTrigger on Opportunity (after insert, before update) {

  Map<String, Id> typeMap = New Map<ID,RecordType>();
  for(RecordType rt: [Select ID, DeveloperName From RecordType Where sObjectType = 'Opportunity']) {
    typeMap.put(rt.DeveloperName, rt.id);
  }

  for (Opportunity opp : trigger.new)  {
      // And the Agreement Category on the record = TEST
      if (opp.Stage__c == 'Negotiation/Review') {
        // Then automatically change the Record Type to ReadOnly RecordType
        opp.RecordTypeID = typeMap.get('ReadOnly');
      }
  }
}
```

view raw

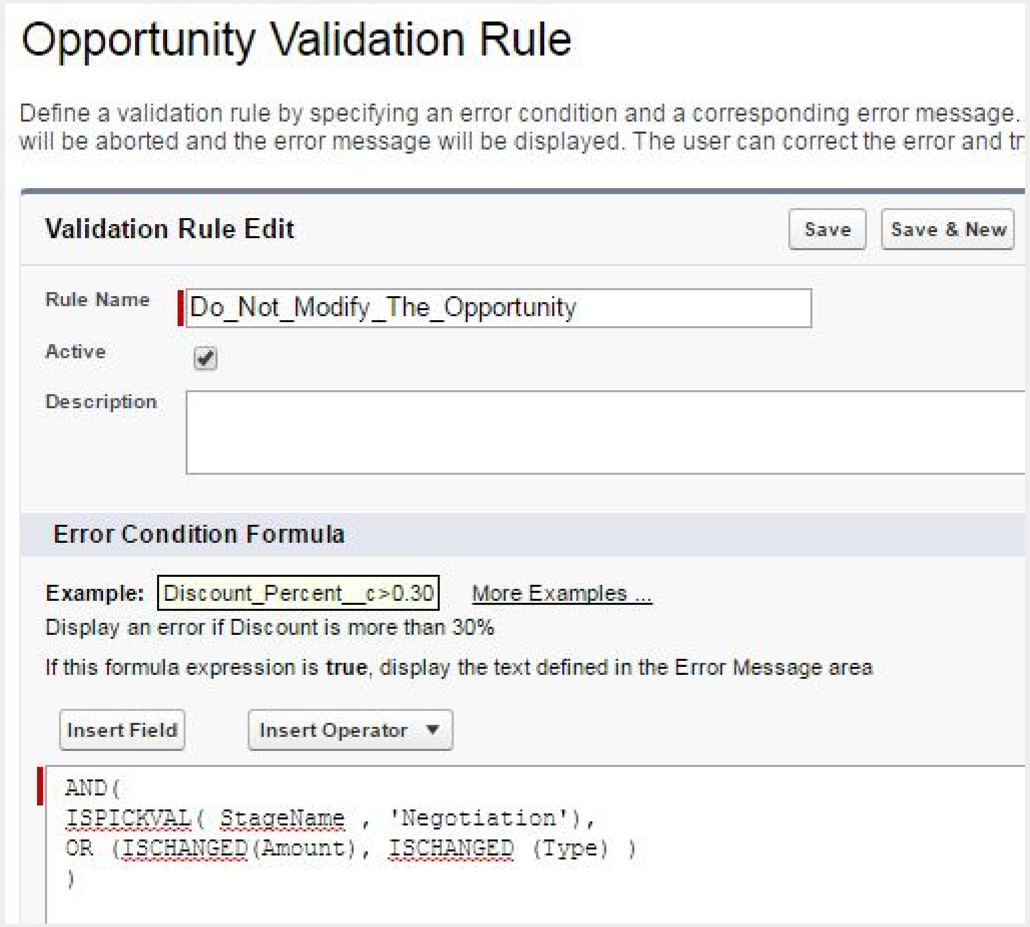
TriggerToChangeRecordType
hosted with ❤ by GitHub

Note: The 2nd approach should be used only when you do not have an option to implement a workflow for any reason. If you are going with Record Types approach, it is strongly recommended to use Approach 1.

3. Without Record Types – Validation Rule

If you prefer to not use Record Types, then you can go with approach #3 OR #4.

In this approach, we will not use any page layouts, record types, workflows or process builder. This is simply by using a Validation Rule on Opportunity object. Here's a validation rule that I created, which checks for the Negotiation Opportunity stage and if the fields Amount or Type are changed.



Opportunity Validation Rule

Define a validation rule by specifying an error condition and a corresponding error message.
will be aborted and the error message will be displayed. The user can correct the error and tr

Validation Rule Edit

Rule Name: Do_Not_Modify_The_Opportunity

Active: ✓

Description:

Error Condition Formula

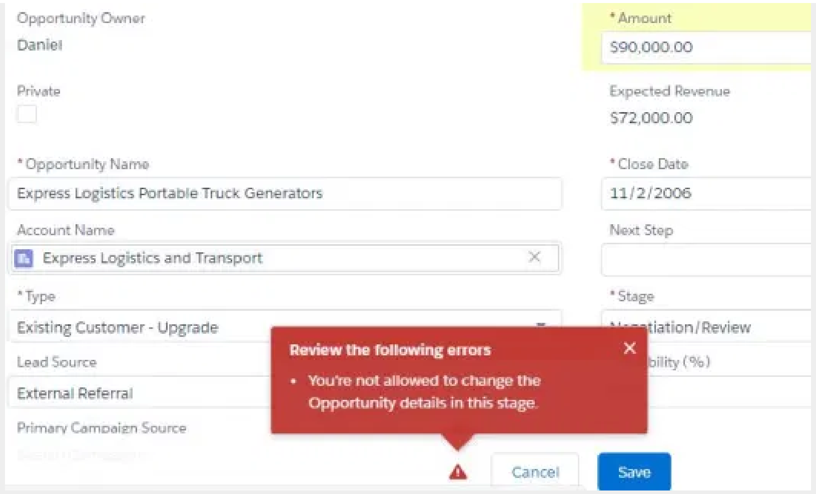
Example: Discount_Percent__c>0.30 More Examples ...

Display an error if Discount is more than 30%

If this formula expression is **true**, display the text defined in the Error Message area

```
AND(
ISPICKVAL( StageName , 'Negotiation'),
OR (ISCHANGED(Amount), ISCHANGED (Type) )
)
```

If any of these 2 fields are changed, the validation rule fires, gives an error message and it won't allow the user to save the opportunity as shown below.


Opportunity Owner
Daniel
*Amount
$90,000.00
Private
Expected Revenue
$72,000.00
*Opportunity Name
Express Logistics Portable Truck Generators
*Close Date
11/2/2006
Account Name
Express Logistics and Transport
Next Step
*Type
Existing Customer - Upgrade
*Stage
Negotiation/Review
Lead Source
External Referral
Primary Campaign Source

Review the following errors
- You're not allowed to change the Opportunity details in this stage.

This is one way to lock the record but note that this in this example I just blocked Amount and Type fields only (mentioned in the validation rule). To block the entire record, you need to add all the ISCHANGED condition for all the fields in the validation rule.

With this approach, even System Administrator won't be able to edit the record. You can specify for which Profiles this validation rule apply by mentioning the Profile Name in the validation Rule.

Disadvantage with this approach is that the error message is shown only when the user clicks on Save. The user will not know that the record is locked for editing until all the changes are made and clicks on Save.

Workaround for this disadvantage is to use an inline Visualforce page which shows the message – "Record Locked". (Using apex outputText with rendered condition as the negotiation stage) so the users are aware that the record is locked. (Even if they edit and try to save the record, the validation rule will not allow to save!)

Note: There is no way to lock the entire record with Validation rule in a single statement. You have to check ISCHANGED condition for all the fields in the validation rule.

4. Without Record Types – Trigger

This approach is exactly same as the 3rd one except that instead of Validation rule, we use a Trigger. Only the validation rule check is done in the trigger. All the disadvantages mentioned for the 3rd approach apply here too.

Note: Ideally if you are not choosing approach 1 because you do not want to use Record Types, then you have to go with Approach 3 (Validation Rule). Only if you run out of limit of number of validation rules on an object, you should go with Approach 4.

5. Visualforce Page

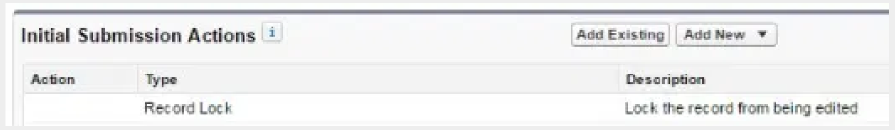
This is a more tedious way to lock a record. You need to override the standard buttons Edit and View with a visualforce page. The visualforce page can use tag to replicate the entire page layout (and disable Edit buttons on the page) OR use individual tags for the fields to build the page from scratch. No user would be able to edit outputField values.

Note: The disadvantage with this approach is maintaining the VF page. If you'd want to add fields to the page layout, you need to edit the VF page and add the code for every new fields. Quite tedious!

6. Approval Process

Another way to lock a record is using Approval Process. The locking feature is a standard feature provided by Salesforce and the entire record will be locked. Well, the whole purpose of Approval Process isn't about locking, it is about how records are approved in Salesforce. An approval process specifies each step of approval, including who to request approval from and what to do at each point of the process. We are just using an Action from the Approval Process.

So, how to lock a record using approval process? Simple, you just need to setup an Approval Process for an object and set the criteria for the record to enter the approval process (In our example, the criteria is Opportunity stage is 'Negotiation'). Once you set the initial criteria, the Initial Submission Action will include **Record Lock.** That's it.


Initial Submission Actions

Action	Type	Description
	Record Lock	Lock the record from being edited

You can set the record edit ability settings to 'Administrators' only OR currently assigned approver.

Note: Implementing the approval process, implies that the record needs to be approved by a user (configurable in the approval process). Even after the record is approved, you can set the record to remain in the Locked state if you prefer to do so OR unlock after approval.