ORACLE NETSUITE

SuiteFlow User Guide

2023.1

June 7, 2023

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document may change and remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://www.oracle.com/corporate/accessibility.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Sample Code

Oracle may provide sample code in SuiteAnswers, the Help Center, User Guides, or elsewhere through help links. All such sample code is provided "as is" and “as available”, for use only with an authorized NetSuite Service account, and is made available as a SuiteCloud Technology subject to the SuiteCloud Terms of Service at www.netsuite.com/tos.

Oracle may modify or remove sample code at any time without notice.

No Excessive Use of the Service

As the Service is a multi-tenant service offering on shared databases, Customer may not use the Service in excess of limits or thresholds that Oracle considers commercially reasonable for the Service. If Oracle reasonably concludes that a Customer’s use is excessive and/or will cause immediate or ongoing performance issues for one or more of Oracle’s other customers, Oracle may slow down or throttle Customer’s excess use until such time that Customer’s use stays within reasonable limits. If Customer’s particular usage pattern requires a higher limit or threshold, then the Customer should procure a subscription to the Service that accommodates a higher limit and/or threshold that more effectively aligns with the Customer’s actual usage pattern.

Beta Features

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Send Us Your Feedback

We'd like to hear your feedback on this document.

Answering the following questions will help us improve our help content:

- Did you find the information you needed? If not, what was missing?
- Did you find any errors?
- Is the information clear?
- Are the examples correct?
- Do you need more examples?
- What did you like most about this document?

Click here to send us your comments. If possible, please provide a page number or section title to identify the content you're describing.

To report software issues, contact NetSuite Customer Support.

Table of Contents

SuiteFlow Overview

Use SuiteFlow to create and execute workflows in NetSuite. A workflow is the definition of a custom business process for a standard or custom record in NetSuite. Business processes can include transaction approval, lead nurturing, and record management. A workflow defines and automates the business process.

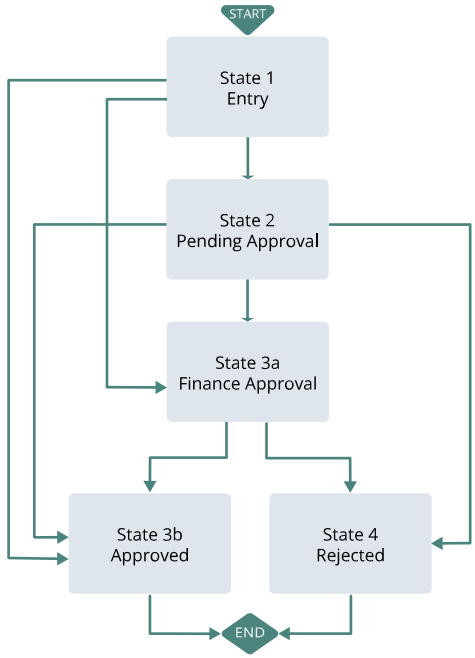
To view, create, and edit workflows, you must have the appropriate permission and permission level required for working with the base record types in the workflow. For access to all SuiteFlow functionality, use the Administrator role. For more information, see Required Permissions for SuiteFlow.

You define workflows for a specific record type and contain the stages, or states, of a record as it moves through the business process. In each state, a workflow defines the actions to be performed, like sending emails or adding buttons to a record form, before the workflow completes or transitions to another state. A workflow can move between different states, or transition, depending on the business process requirements. The actions and transitions can contain conditions that must be met before the action or transitions execute.

NetSuite starts an instance of a workflow on a record and a record transitions between states in a workflow based on specific triggers. Triggers are events that occur when records are viewed, created, or updated. You can also direct NetSuite to run workflow instances on records based on a schedule.

Use the Workflow Manager Interface interface in SuiteFlow to create and edit workflows. Create workflows in the Workflow Manager and a workflow instance initiates and executes according to the defined business process.

The following diagram shows a sample approval business process for an estimate:



In the following example, a sales rep creates an Estimate record. SuiteFlow initiates an instance of the approval workflow to automate the approval process for the estimate. Workflow actions and conditions on each state determine how the estimate transitions through the approval process.

The following table describes the states in the workflow and their associated actions and transitions when the sales rep creates the record:

State	Description
State 1 Entry	The first state a record enters in the workflow after estimate creation is State 1 Entry. This state begins the approval process. The Entry state sets the **Approval Status** field of the estimate to **Pending Approval** and then executes transitions according to the following conditions: ■ If the sales rep has a supervisor, the estimate transitions to State 2 Pending Approval. ■ If the sales rep has no supervisor and the estimate value is less than $50,000, the estimate transitions to State 3b Approved. ■ If the sales rep has no supervisor and the estimate value is $50,000 or greater, the estimate transitions to State 3a Finance Approval.
State 2 Pending Approval	This state locks the record to anyone other than the sales rep's supervisor and adds **Approve** and **Reject** buttons to the record form. Each button is set up to transition to a different state. The next transition depends on the button clicked: ■ Approve. If the estimate value is less than $50,000, record transitions to State 3b Approved. If the estimate value is $50,000 or greater, the estimate transitions to State 3a Finance Approval. ■ Reject. The estimate transitions to State 4 Rejected.
State 3a Finance Approval	This state locks the record to anyone other than the Finance Manager and adds **Approve** and **Reject** buttons to the record form. The next transition depends on the button clicked: ■ Approve. The estimate transitions to State 3b Approved. ■ Reject. The estimate transitions to State 4 Rejected.
State 3b Approved	Sets the **Approval Status** field of the estimate to **Approved** and the workflow completes.
State 4 Rejected	Sets the **Approval Status** field of the estimate to **Rejected** and the workflow completes.

Getting Information About SuiteFlow

The following table describes SuiteFlow concepts and where you can get more information:

Concept	Description
Workflow Manager interface	Use the Workflow Manager interface to create and edit workflows. The interface includes the workflow definition page, diagrammer, and context panel. See Workflow Manager Interface.
Workflow elements overview	Overview of each workflow element and links to more information about using the element. See Workflow Elements.
Workflow audience	The types of users who can create or run workflows. See Workflow Audience.
Workflow initiation	Definition for when NetSuite starts a workflow instance on a record. See Workflow Initiation.
Tutorial	Follow the steps in the tutorial to create a workflow based on an Opportunity record. Use the steps in the tutorial to become familiar with creating the basic elements of a workflow, including states, actions, transitions, and conditions. See Creating Your First Workflow.

Concept	Description
Creating and editing workflows and workflow elements	All the procedures and description of the options required to create a workflow. See Working with Workflows.
Administering workflows	After you create a workflow, you can perform specific administration tasks. These tasks include Searching for workflows, canceling instances, performing mass updates, and bundling workflows. See Workflow Administration.
Reference information	Get more information about the triggers and actions for workflow. Includes detailed information about each server and client trigger and each action types that you can use in a workflow, in addition to examples of use for workflow elements. See SuiteFlow Reference and Examples.
Testing and troubleshooting workflows	During workflow development, you need to test your workflows. NetSuite tracks workflow instance activity and generates execution logs to assist you with workflow testing and troubleshooting. See Testing and Troubleshooting Workflows.
Example workflows	Use the examples to see how to create certain types of workflows. Each example includes the detailed steps required to create and then test the workflow. See Workflow Samples.

Accessing SuiteFlow

To access the SuiteFlow UI, you need to have the SuiteFlow feature enabled in your NetSuite account. You also need to use a role that has the Workflow permission assigned to it. For more information about enabling the SuiteFlow feature, see Enabling SuiteFlow. For more information about SuiteFlow permissions, see Required Permissions for SuiteFlow.

To access the SuiteFlow UI, go to Customization > Workflow > Workflows. To create a new workflow, click **New Workflow**.

Required Permissions for SuiteFlow

Access to NetSuite data and to the NetSuite UI is based on users, roles, and permissions. Each role includes a set of associated permissions that determine the data users can see and the tasks that they can perform. Permissions are associated with roles and roles are assigned to users. For more information about roles and permissions, see the help topic NetSuite Users & Roles.

For access to all SuiteFlow functionality, use the Administrator role. For more information about the Administrator role, see the help topic The Administrator Role.

The following table describes the permissions related to SuiteFlow, the associated tasks, and any additional requirements.

Permission Name	Usage Description	Notes
Workflow	View, create, and edit workflows	■ SuiteFlow must be enabled in your account ■ You must have the appropriate permission and permission level for working with the base record types in the workflow
SuiteScript	Create and edit workflow action script records	■ SuiteFlow must be enabled in your account

Permission Name	Usage Description	Notes
Custom Fields	■ View and edit the Workflow Custom Field record ■ View and edit the Workflow State Custom Field record	—
Core Administration Permissions	Use the Cancel field in Workflow Instance searches	—
Control SuiteScript and Workflow Triggers in Web Service Request	Set the SOAP header preference for disabling scripts and workflow triggers per request	—
Control SuiteScript and Workflow Triggers per CSV Import	Check or clear the run Server SuiteScript and Trigger Workflow box for each report	—

To add the SuiteFlow permission to a standard role, go to Setup > Users/Roles > Manage Roles (Administrator). On the Manage Roles page, click **Customize** next to the role that you want to add the SuiteFlow permission to. On the **Permissions** subtab, click **Setup**. In the **Permissions** dropdown list, click **Workflow**, and then click **Add**. Click **Save** to apply your changes to the role.

Enabling SuiteFlow

Access to SuiteFlow is controlled using roles and permissions. You need to be logged into and using a role with the Workflow permission assigned to it. Additionally, the SuiteFlow feature must be enabled on the account you are using.

Note: To enable SuiteFlow on an account, you must be logged into the Administrator role.

To enable SuiteFlow:

1. Go to Setup > Company> Setup Tasks > Enable Features.
2. Click the **SuiteCloud** tab.
3. Under SuiteFlow, check the **SuiteFlow** box.
4. Click **Save**.

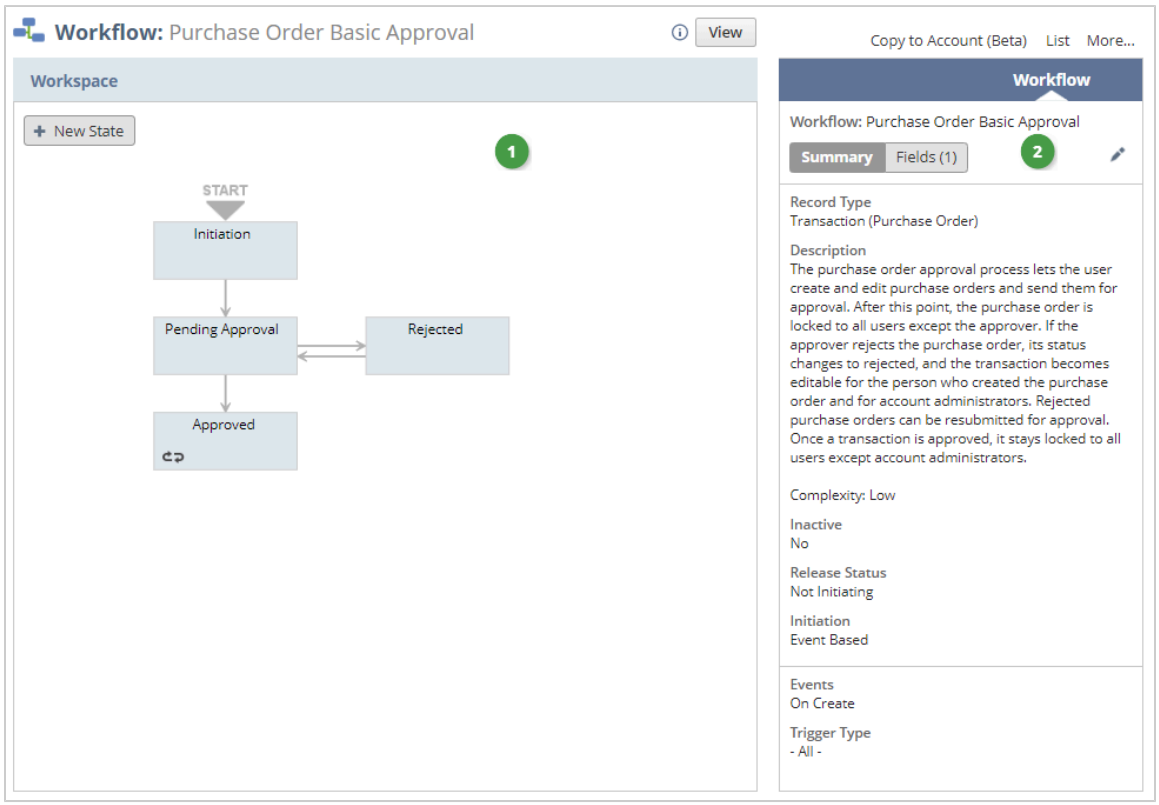
Important: When you enable SuiteFlow, you may be guided through a series of prompts to enable Client and Server SuiteScript. Do not turn off any of these features after they are enabled.

After SuiteFlow is enabled on an account, users with the Workflow permission can view, create, and edit workflows. For more information about workflow permissions, see Required Permissions for SuiteFlow.

Workflow Manager Interface

The Workflow Manager is a drag-and-drop tool that you use to build workflows in NetSuite. Use the Workflow Manager to create, view, and edit the state, action, and transition workflow elements.

The following screenshot shows the Workflow Manager interface in edit mode:



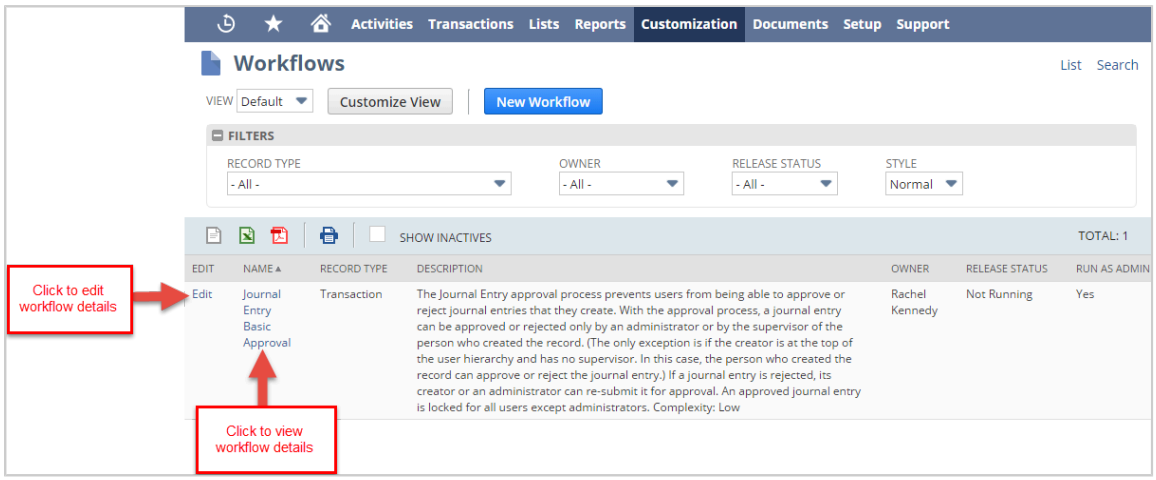
The Workflow Manager includes the following interface elements:

- 1. **Diagrammer.** In edit mode, use the diagrammer to add and edit states and transitions. In view mode, user the diagrammer to view workflow states, actions, and transitions. See Workflow Diagrammer.
- 2. **Context Panel.** In edit mode, use the context panel to add, edit, and delete workflow elements. In view mode, use the context panel to view workflow element information. See Workflow Context Panel.

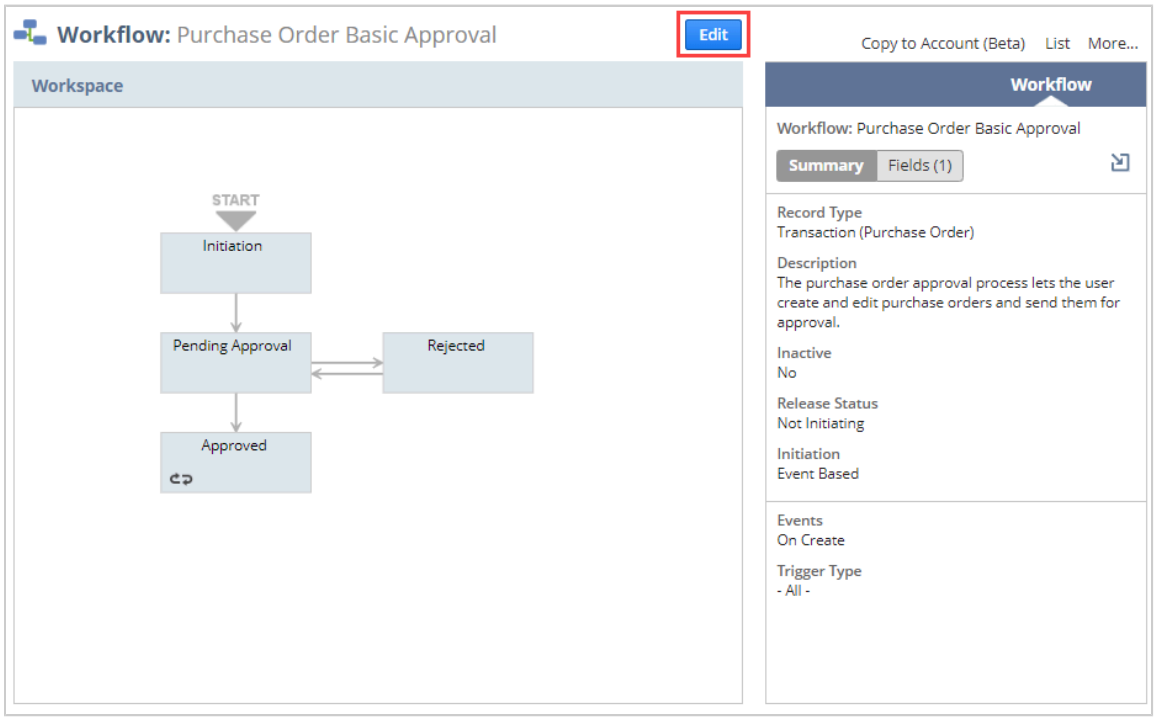
For more information about view and edit modes, see Workflow Manager Interface View and Edit Modes.

Workflow Manager Interface View and Edit Modes

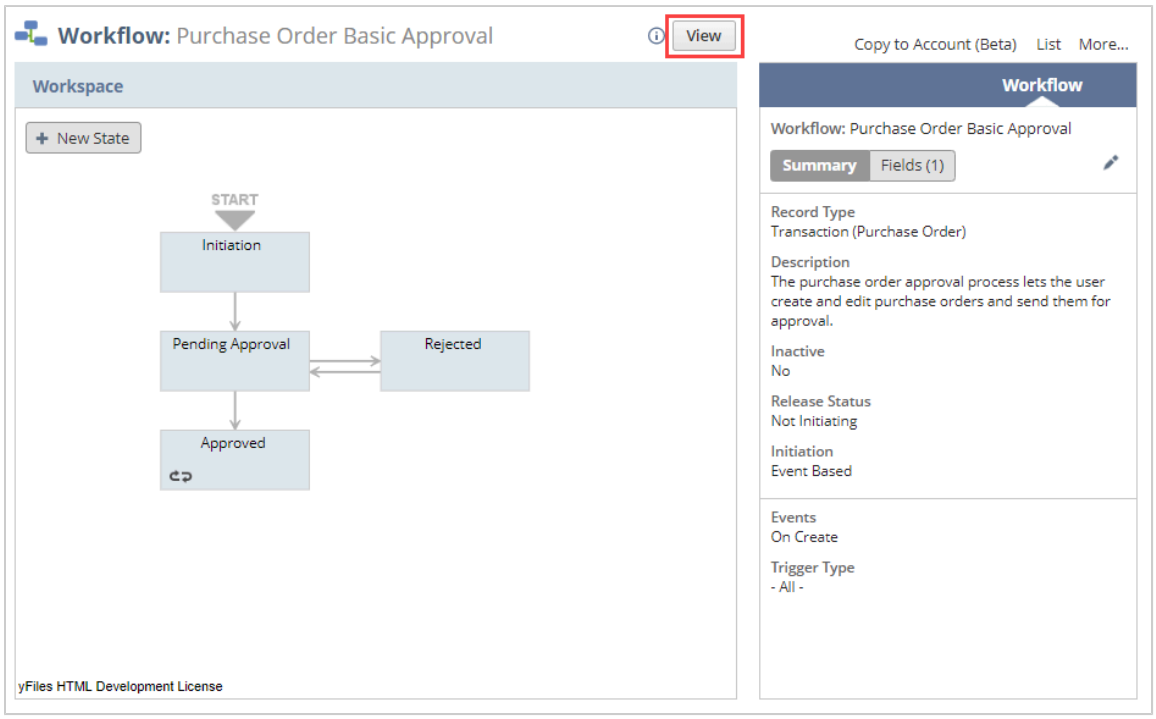
The Workflow Manager Interface has two modes: view and edit. Use view mode to view workflow details. Use edit mode to configure workflow details. Access view mode from the Workflow list page (Customization > Scripting > Workflows) by clicking a workflow name. Access edit mode also from the Workflows list page by clicking **Edit** to the left of a workflow name.



In view mode, click **Edit** to switch to edit mode.

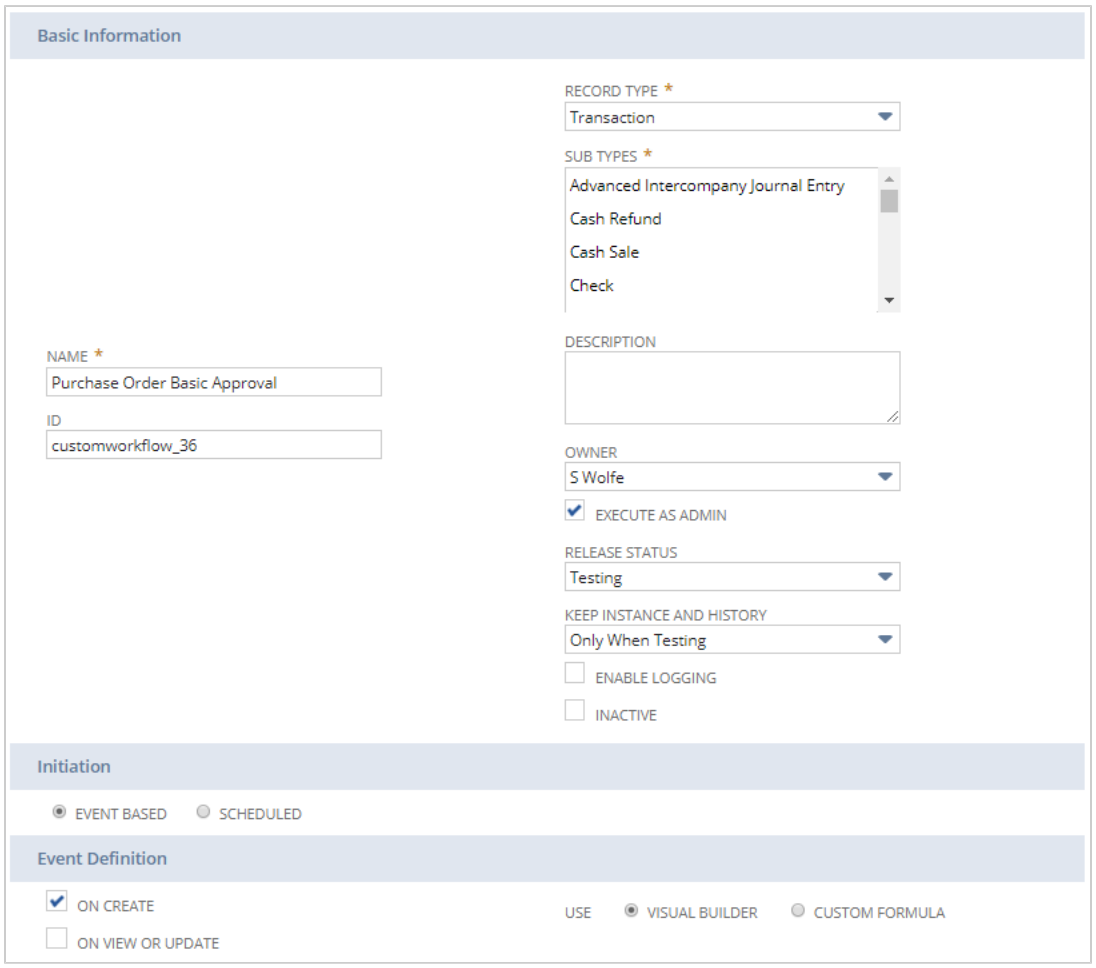


In edit mode, click **View** when you are finished making modifications to the workflow and to return to view mode.



Workflow Definition Page

The workflow definition page contains the properties of a workflow.



The following table describes the type of properties that you set on the workflow definition page when in edit mode:

Property Type	Description
Basic properties	Basic properties of all workflows include the following properties: ■ **Record type and sub type.** All workflows must have a base record type. For Customer, Item, and Transaction records, you can also specify a sub type. See Workflow Base Record. ■ **Owner.** The owner of the workflows determines for whom the workflow will run during testing. The owner also receives emails for errors that occur during workflow execution. For information, see Workflow Audience. ■ **Execute as Admin.** Indicates that the workflow runs as a user with administrator permissions. See Creating a Workflow. ■ **Release Status**. Indicates the workflow's release status. Options include Suspended, Not Initiating, Testing, and Released. The default option is Testing. See the help topic Release Status. ■ **Keep Instance and History**. Specify when workflow instance, workflow history, and workflow logs are saved for a workflow. Options include Only When Testing, Never, and Always. See Disabling History for a Workflow. ■ **Enable logging.** Specify whether the workflow generates a workflow execution log when the workflow runs. See Creating a Workflow. ■ **Inactive.** Indicates the workflow does not initiate at any time. See Inactivating a Workflow.

Property Type	Description
Initiation	Workflow initiation includes the following options: ■ **Event Based.** The workflow initiates on a record create event, record update event, or record view event for the base record type. For information, see Initiating a Workflow on an Event. ■ **Scheduled.** Run a workflow on a schedule. For information, see Initiating a Workflow on a Schedule.
Event Definition	Define the events and conditions for workflow initiation. Includes the following properties: ■ **On Create or On Update.** Indicates the workflow initiates when a record is created, updated, or viewed. For information, see Initiating a Workflow on an Event. ■ **Trigger type.** The server trigger on which the workflow initiates. For information, see Server Triggers. ■ **Event Types** and **Contexts**. You can restrict the workflow initiation to specific event and execution context types. For information, see Workflow Event Types and Execution Contexts. ■ **Condition.** Use a condition or custom formula to define requirements that must be met for the workflow to initiate. For information, see Workflow Conditions. ■ **Saved search.** Use a saved search as a workflow initiation condition.

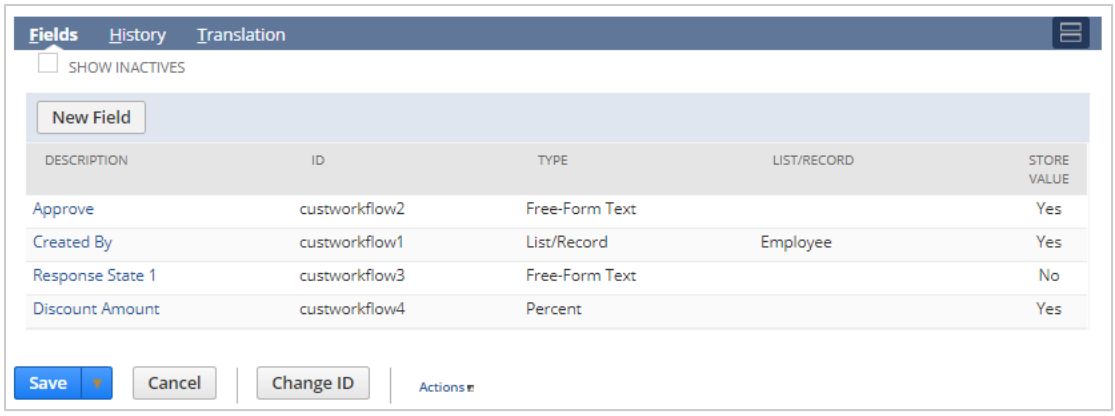
At the bottom of the Workflow Definition page there are two subtabs:

- Workflow Definition Page Fields Subtab: used to create custom workflow fields. For more information, see Workflow Definition Page Fields Subtab.
- Workflow Definition Page History Subtab: used to view a historical list of changes made to the workflow. For more information, see Workflow Definition Page History Subtab.

Note: For more information about using this page to create or edit a workflow, see Working with Workflows.

Workflow Definition Page Fields Subtab

The Fields subtab on the Workflow Definition page lists the fields created for the workflow. Use this tab to view and create workflow fields.



To create a workflow field from the Workflow Definition page Fields subtab:

1. If you have not already done so, open the workflow you want to create a field in at Customization > Scripting > Workflows.

2. On the Workflow Context Panel, click the edit icon to open the Workflow Definition page. The Fields subtab displays at the bottom of the page by default.
3. Click **New Field**.
4. For detailed information about creating a workflow field, see Creating and Using Workflow Fields.

Workflow Definition Page History Subtab

This subtab provides historical log information about changes made to the workflow. Access the History subtab from the Workflow Context Panel by clicking the edit icon and selecting the History subtab at the bottom of the page.

Note that errors are not logged on the History subtab. This is because when an error occurs during a workflow's execution, the workflow is rolled back and the instance of the workflow no longer exists in the database. As such, errors cannot be logged for an instance of a workflow that no longer exists in the database.

The following table describes the default columns displayed on the Workflow History subtab:

Column	Description
Date/Time	The date and time when the change took place.
User	The user who made the change.
Component	The workflow element that was changed, for example, Workflow, Transition, or Action.
Script ID	The script ID of the workflow element that was changed.
Type	The type of change that was made: Create, Edit, or Delete.
Note	Includes information about the following changes: ■ Indicates if a workflow has changed to Active or Inactive on the Workflows list page. ■ Indicates when a workflow's release status is changed on the Change Status page for locked workflows, for example, changed from Testing to Released. ■ Indicates if a workflow has been copied. ■ Indicates the updated workflow release status. ■ Indicates workflow script ID changes.
Revision	The number of the workflow revision. Any change made to a workflow's definition or any of its children increases its revision number. For more information, see Workflow Revisions.

The following table describes information logged on the tab:

Workflow Element	Logged Actions
Workflow	■ When a workflow is created or updated on the Workflow Definition page. ■ When a workflow is set to Active or Inactive on the Workflows list page. ■ When a workflow's release status is changed on the Change Status page for locked workflows, for example, changed from Testing to Released. ■ When a workflow is copied.
State	■ When a state is created, updated, or deleted from the State Definition page. ■ When a state is deleted from the Workflow Diagram.

	■ When a state is deleted from the Context Panel. ■ When a state is moved in the Workflow Diagram.
Action	■ When an action is created, updated, or deleted from the Action Definition page. ■ When an action is deleted from the Context Panel.
Transition	■ When a transition is created, updated, or deleted from the Transition Definition page. ■ When a transition is deleted from the Context Panel. ■ When a transition is deleted from the Workflow Diagram.
Condition	■ When a condition is created, updated, or deleted from the Workflow / Action / Transition Definition pages. ■ When a condition is deleted from the Workflow Condition Definition page.
Script ID	■ When a workflow element's script ID is changed. ■ When a workflow custom field script ID is changed. ■ When a workflow custom state field script ID is changed.

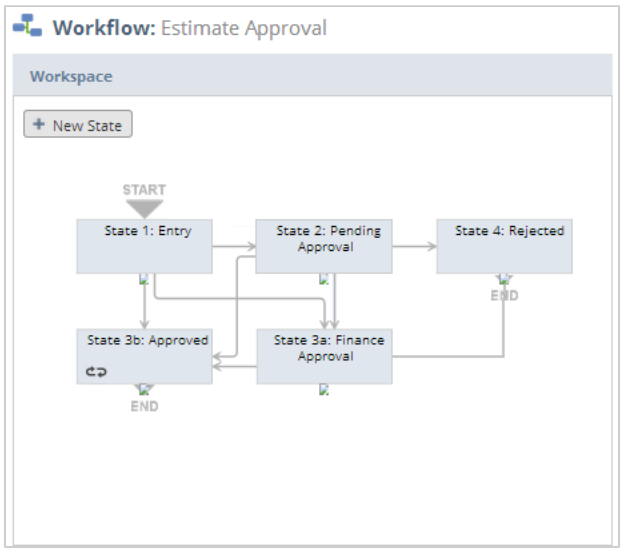
The following table describes several special cases for logging of data on the History subtab:

Workflow Element	**Special Cases**
Workflow	■ Logs are created when a workflow is set to Active or Inactive on the Workflows page. ■ Logs are created when a workflow's status is changed on the status page for locked workflows. For example, a workflow status is changed from Testing to Released. ■ When a workflow element is deleted, only one log entry appears on the History subtab, regardless of any child elements that are also deleted. ■ Workflow deletion is not logged, but you can search for deleted workflow records. See the help topic Searching for Deleted Records. ■ Parent relationship is not logged. For example, when an action is created, the state in which the action is created is not logged. ■ When a workflow definition is edited, each time **Save** is clicked, it is logged on the History subtab. Logs are created even when no change is submitted due to the **Save**.
State	■ When a state is edited, each time **Save** is clicked, it is logged on the History subtab. Logs are created even when no change is submitted due to the **Save**.
Action	■ Custom Action is logged as Action. ■ Action order is logged according to where it was changed: □ Action order changed from the State Details page is logged as State Edit. □ Action order changed from the Action Details page is logged as Action Edit. ■ When an action is edited, each time **Save** is clicked, it is logged on the History subtab. Logs are created even when no change is submitted due to the **Save**.
Action Groups	■ Logged as Group Actions.
Transition	■ When a transition is edited, each time **Save** is clicked, it is logged on the History subtab. Logs are created even when no change is submitted due to the **Save**.
Condition	■ When all lines of a condition are deleted, it is logged as Condition Edit. ■ When a condition is deleted on the Condition Definition page, it is logged as Condition Delete. ■ On the Action Details page, when the **Use** radio button is changed from **Visual Builder** to **Custom Formula**, it is logged as WF/Action Edit.

Workflow Diagrammer

The Workflow Diagrammer is a JavaScript-based editor that you can use to create and edit workflow states and transitions when in edit mode. It includes drag and drop functionality you can use to move states and transitions to reflect the business process defined by the workflow. Any change made in the diagrammer is automatically saved.

The diagrammer also indicates the start and end states for the workflow and indicates states that are set to not exit the workflow. The diagrammer determines a state as an end state if it has no transitions to another state.



You can complete the following tasks with the diagrammer:

- **Create and edit states.** Click the **New State** button to create a state and double-click the state to open the **State** window and edit the state properties and state actions, transitions, and custom fields. For more information about creating and editing states, see Working with States.
- **Create and edit transitions.** Click the icon at the bottom of a state and drag to create a transition to another state. Double-click an existing transition to open the **Transition** window and edit the transition properties. For more information about creating and editing transitions, see Working with Transitions.
- **Reorganize the workflow states and transitions.** Click and drag states to reorder the workflow to properly reflect the business process.

Workflow Context Panel

The context panel displays contextual workflow properties. When in edit mode, select an object in the diagrammer and use the panel to open states, actions, and transitions for edit; create actions; open the workflow definition properties for edit; and create, edit, and delete workflow fields.

The tabs displayed in the context depend on the object selected in the diagrammer:

- **State tab**
  - Open states for edit.
  - Create, edit, and delete actions and actions in groups.
  - Create, edit, and delete state fields.

See State Tab on Context Panel.

- **Transition tab**
  - Open transitions for edit.

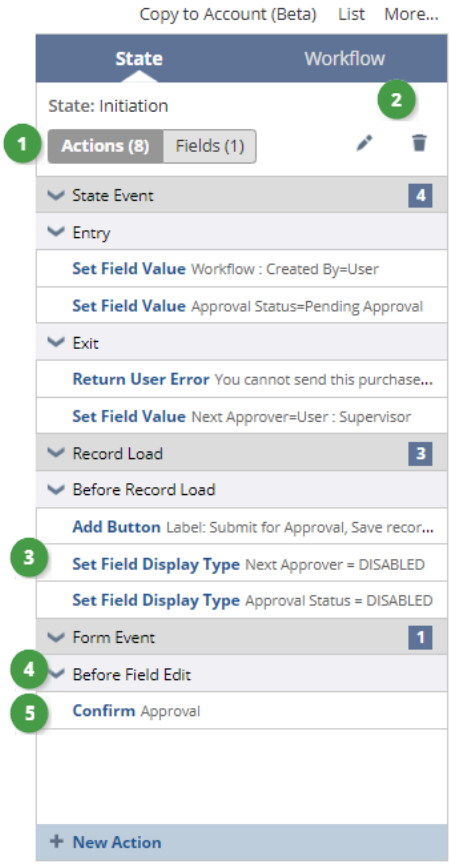
  See Transition Tab on Context Panel.
- **Workflow tab**
  - View workflow definition properties.
  - Open workflow definition properties for edit.
  - Create, edit, and delete workflow fields.

  See Workflow Tab on Context Panel.

State Tab on Context Panel

Use the State tab to work with workflow states, workflow actions, action groups, sublist action groups, and state fields. Switch between the **Actions** and **Fields** views to create, edit, and delete actions and state fields.

The following screenshot shows the State tab:



Use the State tab to complete the following tasks:

- Edit a state to add, edit, or delete actions, transitions, and state fields, or other state properties. Click the **Edit** icon to edit the State properties in the **Workflow State** window. For more information about states and state properties, see Workflow States and Workflow Custom Fields.

- Use **New Action** to create a new action. For more information about actions, see Workflow Actions and Creating an Action.
- View action property summaries. When you point to an action, the context panel displays a summary of the properties for the action.
- View action groups and the actions within action groups. When you point to an action group, icons appear that allow you to add actions to the group, edit the group, or delete the group from the state. For details about action groups, see Using Action Groups.
- View sublist action groups and the actions within sublist action groups. When you point to a sublist action group, icons appear that let you add actions to the group, edit the group, or delete the group from the state. For details about sublist action groups, see Using Sublist Action Groups.

Trigger Categories on the State Tab

Actions that you create display on the State tab under their corresponding server or client trigger category:

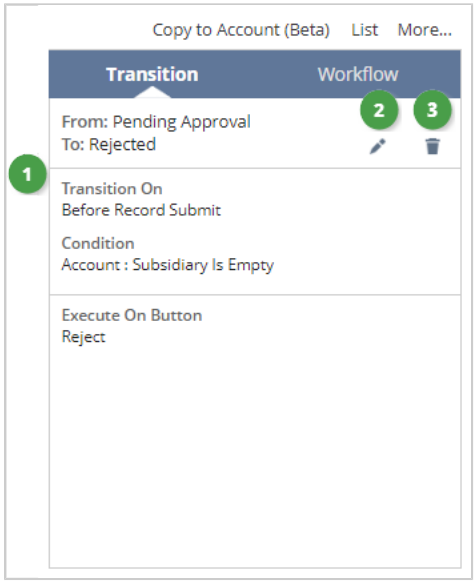
- State Event (when the workflow enters or exits a state)
- Record Load
- Form Event (any client-side user action)
- Record Save
- Scheduled

For more information about trigger execution, see Workflow Triggers.

Transition Tab on Context Panel

Use the Transition tab to edit and delete transitions that you create in the diagrammer or create through the state properties.

The following screenshot shows the Transition tab:



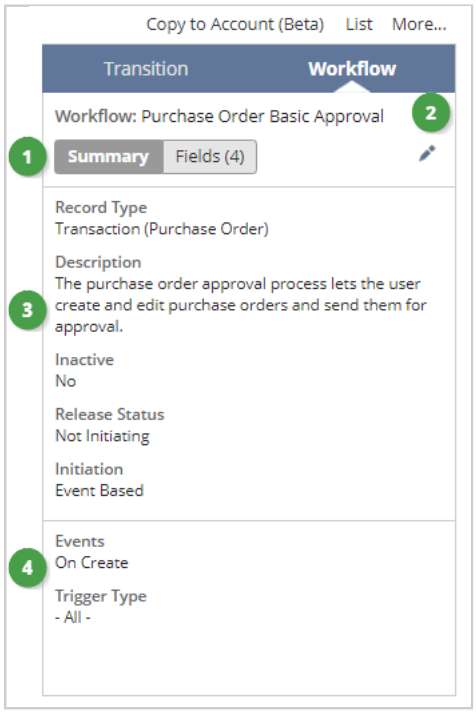
Use the Transition tab to complete the following tasks:

- Edit or delete a transition. Click the **Edit** icon to edit the transition properties in the **Workflow Transition** window. For more information about transitions, see Workflow Transitions. and Creating a Transition.
- View the transition properties that include the **From** and **To** states, the optional transition trigger and the transition condition.

Workflow Tab on Context Panel

Use the Workflow tab to view and edit the workflow properties and custom fields. Switch between the Workflow and Fields views to view a summary of workflow properties, edit the workflow properties, or create, edit, and delete workflow fields.

The following screenshot shows the Workflow tab:



Use the Workflow tab to complete the following tasks:

- Edit workflow properties. Click the **Edit** icon to edit the workflow properties in the **Workflow** window. For more information about workflow properties, see Creating a Workflow.
- View workflow properties. Properties include the record type for the workflow, the initiation type, and the trigger type.
- Create, edit, and delete workflow fields. For more information, see Workflow Custom Fields.

Workflow Elements

The following table describes the elements of a workflow:

Element	Description
Base record	Record type for which you create a workflow. See Workflow Base Record.

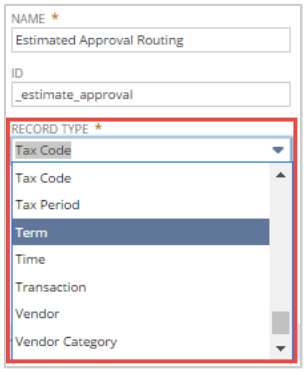
Element	Description
States	Correspond to a stage or status of a record within a specific business process. See Workflow States.
Triggers	Based on events that occur during the processing of a record or user activity on a record. See Workflow Triggers.
Actions	Used to perform specific tasks based on the properties of a record. See Workflow Actions.
Transitions	Moves a record to another state in a workflow. See Workflow Transitions.
Conditions	Requirements that must be met for a workflow to initiate or an action or transition to execute. See Workflow Conditions.
Custom fields	Variables that you can use in workflows, actions, and transitions. See Workflow Custom Fields.
Revisions	The nature and number of modifications made to a workflow's definition or any of its children. See Workflow Revisions.

Workflows can initiate and actions and transitions can execute based on specific event types and context types. See Workflow Event Types and Execution Contexts and Workflows.

Workflow Base Record

The workflow base record is the record type for which you are using a workflow to define a business process. The record type can be a standard NetSuite record type, like a Customer or Transaction, or it can be a custom record type.

Specify the base record type in the **Record Type** dropdown list on the workflow definition page. Any record type for which you can create a workflow appears in the dropdown list. To access a workflow on the Workflows page, you must have full permission for the base record on which the workflow was built. Additionally, you must have the feature related to the base record enabled in your account.



Note: You cannot change the workflow record type after you save the workflow definition.

You can also pick sub types for certain types of records. The following table lists the record types for which you can select a sub type. If you select one of these record types in the **Record Type** dropdown list, the **Sub Type** field becomes required. You must then select at least one sub type before you can save the workflow.

Record Type	Available Sub Types
Customer	■ Customer

Record Type	Available Sub Types
	■ Lead ■ Prospect
Item	Any record of type Item.
Transaction	Any record of type Transaction.

Workflow States

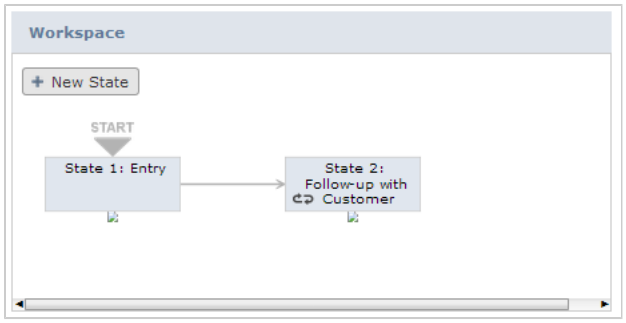
A state corresponds to a stage or status of a record within a specific business process. A workflow can contain as many states as determined by the business process. Every NetSuite workflow must contain at least one state.

Each state can contain actions and links to other states through transitions. When a record enters a state in a workflow, the actions execute according to the action triggers and conditions, and in the order they appear in the state. Then, any transitions to the next state in the workflow execute. Actions and transitions can also contain conditions that determine when or how they execute.

The first state in a workflow is the entry state, or when the record enters the workflow. The entry state in a workflow is indicated with a **Start** icon in the diagrammer and the **Start State** property enabled for the state. A workflow can consist of only a single state, called a single state workflow.

The last state in a workflow is the end state, indicated by the **End** icon in the diagrammer. The diagrammer indicates states with no outgoing transitions as end states. A workflow can have multiple end states. In addition, an end state can be set to not exit the workflow, called a non-exiting state.

The following example shows a sample two state workflow for an Opportunity record, where the workflow does not exit after it enters the second state:



Tip: You can create all of the states in your workflow at one time. You may find it easier to create all workflow states during the initial layout phase and then go back and edit states to add transitions and actions.

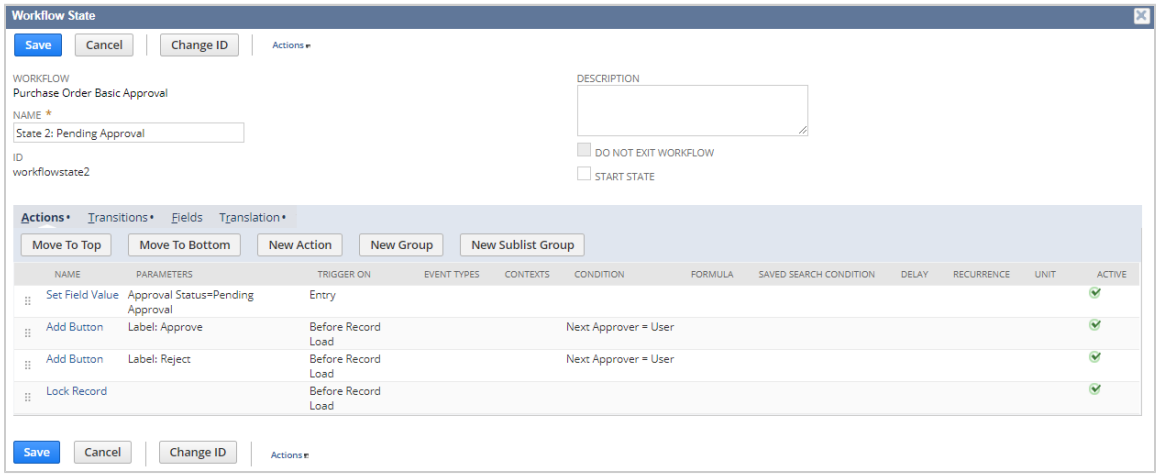
State Properties

You set properties for a state in the **Workflow State** window. You can open this window from the **Edit** icon in the context panel.

You also use the Workflow State window to create and edit the following workflow elements:

- Actions. See Workflow Actions.
- Transitions. See Workflow Transitions.
- State fields. See Workflow Custom Fields.

The following screenshot shows the **Workflow State** window:



More Information about States

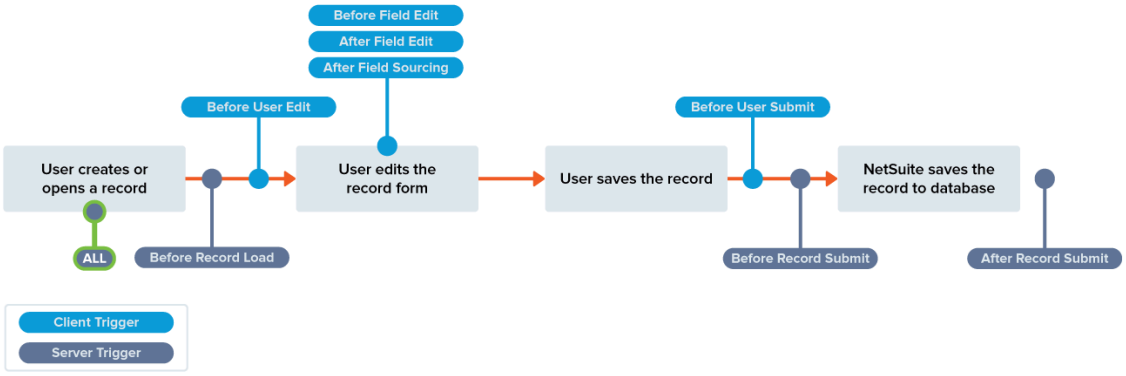
Use the following table to get more information about working with states in a workflow:

Task	For more information
State properties and adding a state to a workflow	Creating a State
Editing a state	Editing a State
Deleting a state	Deleting a State
Creating a state field	Creating and Using State Fields
Working with exit states	Exit States
Creating a non-exiting state	Non-Exiting Workflow States
View state reference information	States Reference

Workflow Triggers

Triggers are based on events that occur during the processing of a record in NetSuite and dictate when the workflow should perform certain tasks. NetSuite defines triggers for when a workflow should initiate, when actions should be performed, or when a record should transition to another state for the record, based on the processing of a record in NetSuite. You direct a workflow to respond to the record processing events.

SuiteFlow includes two types of triggers, server triggers and client triggers:



In general, the type of trigger you use in SuiteFlow depends on when you want a workflow to either initiate or complete a specific action or transition. For example, if you want to alert users that they must change a field value before they submit the record, issue an alert on a client trigger. If you want to notify users after they submit the record with a system error, issue an alert on a server trigger. You can only initiate workflows on server triggers.

Limiting the actions that a workflow performs can improve its layout. Likewise, choosing the correct type of trigger optimizes workflow execution and reduces the potential for missing information. This is especially important for workflows that execute on a client trigger, as they involve the user interacting with the interface.

You can initiate a workflow or transition to another state on any of the server triggers or set up actions to execute on the server and client triggers. The triggers supported for actions depend on the type of action. To understand the order in which triggers execute for actions or transitions in a workflow state, see the SuiteFlow Trigger Execution Model.

Server Triggers

Server triggers occur when a record is read from or written to the NetSuite database or when a record enters or exits a state in a workflow. Server triggers are classified as either record based, workflow based, or time based.

- Record based triggers occur when something happens to the record:
  - The record is going to load (Before Record Load trigger)
  - After the record loads, but before it is saved (Before Record Submit trigger)
  - The record is saved (After Record Submit trigger)
- Workflow based triggers occur when something happens to the workflow
  - A state has been entered (On Entry trigger)
  - A stated has been exited (On Exit trigger)
- Time based triggers occur according to a user-defined schedule (Scheduled trigger)

Workflows evaluate and execute all actions in a state for a specific server trigger before they evaluate and execute the transitions for that trigger.

The following table describes the server triggers and the type of workflow tasks to which they apply:

Trigger Name	Applies To	Description
Before Record Load	■ Workflow initiation	Triggers when a new record is created, before it is loaded into the browser, or before an existing record loads into the browser.

Trigger Name	Applies To	Description
	■ Actions ■ Transitions	■ **Actions.** Execute every time a record is loaded in a state. ■ **Transitions.** Execute when a user loads a record. Use this trigger, for example, when setting default values on fields for the record form, hiding fields on the record form, or locking a record.
Before Record Submit	■ Workflow initiation ■ Actions ■ Transitions	Occurs after a user clicks **Save** on a record and before NetSuite saves the record data to the database. ■ **Actions.** Execute every time a user clicks **Save** when a record is in a state. ■ **Transitions.** Execute after a user clicks **Save**. Use this trigger, for example, when validating record form fields or calculating field values before saving a record.
After Record Submit	■ Workflow initiation ■ Actions ■ Transitions	Occurs after NetSuite saves the record data to the database. ■ **Actions.** Execute after a record is saved to the database when a record is in a state. ■ **Transitions.** Execute after a record is saved to the database. Use this trigger, for example, when sending an email that a record has been changed or when creating dependent records.
Scheduled	■ Workflow initiation ■ Actions ■ Transitions	You can create a schedule for workflows to initiate, and for actions and transitions to execute.
Entry	■ Actions ■ Transitions	Occurs the first time that a workflow enters a state at the same time as the first server trigger*. ■ **Actions.** Execute the first time a record enters a state. Workflows can enter a state multiple times. ■ **Transitions.** Execute when a record enters a state. *On Entry triggers do not execute on their own. They execute at the same time as the first ordered server trigger for a state, for example, the On Entry trigger executes when the workflow enters a state simultaneously with the Before Record Load trigger.
Exit	■ Actions	Occurs when a workflow exits a state and transitions to another state.
ALL	■ Workflow initiation	For workflow initiation only. The workflow initiates on any triggering event.

Note: The Scheduled trigger is considered a server trigger. However, scheduled actions and transitions are not directly associated with user events such as loading or saving a record. Scheduled actions and transitions are automatically initiated by NetSuite, based on user-defined time increments. For more information, see Scheduled Trigger and Initiating a Workflow on a Schedule.

Client Triggers

Client triggers execute when a user interacts with a record form in NetSuite. You can view the client triggers used for actions for a specific workflow state under **Form Event** on the **State** subtab of the context panel.

The following table describes the client triggers:

Trigger Name	Description
Before User Edit	Executes when the record form loads into the browser. Use this trigger, for example, to make changes to the record form in the browser before the user edits any field.
Before Field Edit	Executes when user tabs or clicks away from a field after entering a value. Use this trigger, for example, when validating the value of a record field.
After Field Edit	Executes when a user enters or changes the value of a field. Use this trigger, for example, when dynamically updating the values of other fields when a user changes the value of a specific field.
After Field Sourcing	Executes after a field change, after all of the child field values for the field are sourced. Use this trigger, for example, when setting field values based on other sourced values.
Before User Submit	Executes every time a user clicks **Save** when the form is in the state. The actions execute in the browser, before any data is sent to the NetSuite database and the save operation occurs. Use this trigger, for example, when validating record form field values.

More Information About Server and Client Triggers

Use the following table to get more information about working with server and client triggers:

Task	For more information
Understanding the SuiteFlow trigger execution model	SuiteFlow Trigger Execution Model
Get a description of each trigger type, including examples	Triggers Reference
View which triggers did or did not execute	Workflow Execution Log
Using triggers for workflow initiation	Workflow Initiation
Using triggers for actions	Action Triggers
Using triggers for transitions	Transition Triggers

SuiteFlow Trigger Execution Model

When a record in a workflow enters a state, the workflow instance executes actions and transitions in the following order:

1. Actions and transitions with an Entry trigger execute. The actions only execute the first time a record enters a state. Then the transitions execute.
2. Actions for the trigger on which the record entered the state execute.
3. Transitions for the trigger on which the record entered the state are evaluated and executed, if applicable.
4. If the record transitions to another state, actions with a trigger of Exit execute before the record transitions to the next state.

The following table describes the general order of trigger execution for actions and transitions for a state in a workflow, depending on the record action when the record is in the state and the trigger on which the record entered the state.

Record Action	Trigger	Actions	Transitions*
User creates or opens a record.	Before Record Load	1. Actions set to trigger on Entry execute the first time the record enters the state. 2. Actions set to trigger on Before Record Load execute.	1. Transitions set to trigger on Entry execute. 2. Transitions set to trigger on Before Record Load execute.
Record loads into the browser.	Any client trigger	Actions set to a client trigger execute, if applicable.	None. Transitions do not execute on client triggers.
User clicks **Save** on the record.	Before Record Submit	Actions set to trigger on Before Record Submit execute.	Transitions set to trigger on Before Record Submit execute.
	After Record Submit	Actions set to trigger on After Record Submit execute.	Transitions set to trigger on After Record Submit execute.
*** Note:** Transitions with no **Trigger On** value (blank trigger) execute as soon as the conditions are met.			

For example, if the record enters the state on the Before Record Load trigger, actions set to trigger on Entry and all the subsequent triggers may execute, depending on the state layout.

However, if the record enters the state on the Before Record Submit trigger, only actions set to trigger on Entry, Before Record Submit, and After Record Submit execute. Actions and transitions for the Before Record Load trigger and any client triggers do not execute. For more information, see Initiating or Entry Trigger Rules.

Initiating or Entry Trigger Rules

Understanding the SuiteFlow trigger execution model depends on knowing the trigger on which a workflow instance initiates and the trigger on which a record enters a state:

- **Trigger on which a workflow initiates.** When a workflow initiates, it begins executing the server triggers in the following order, starting with the trigger on which the workflow initiated:
  - Before Record Load
  - Before Record Submit
  - After Record Submit

  If a workflow initiates on an ALL or Before Record Load trigger, the workflow instance executes the Before Record Load trigger first, then the Before Record Submit and After Record Submit triggers after the user saves the record. If a workflow instance initiates on a Before Record Submit trigger, the Before Record Load trigger never executes.

  For more information about initiating a workflow, see Workflow Initiation.
- **Trigger on which a workflow enters a state.** The trigger on which a record enters a state depends on the server trigger that the workflow instance is currently executing.

  For example, a workflow instance executes actions in a state for the Before Record Submit trigger and then transitions to another state. The current trigger being executed is Before Record Submit. Consequently, the workflow instance enters the next state on a Before Record Submit trigger. Any actions or transitions in the next state that trigger on a Before Record Load trigger do not execute.

  You can view these triggers and their order of execution in the Workflow Execution Log.

Workflow Trigger Execution Example

You create a workflow to mark a sales order field as required, validate the field value, and then send an email to the user's supervisor after the record is saved. You create a workflow that initiates on the creation of a sales order record and triggers on the Before Record Load server trigger.

The following table describes the states in the workflow:

State Name	Description
State 1 Entry	Contains an action and a transition. The action triggers on Entry and makes a field required on the record form. The transition triggers on Before Record Load and transitions to State 2.
State 2 Field Validation	Contains an action to validate the value of the field when the user modifies it. Contains a transition that triggers on Before Record Submit and transitions to State 3.
State 3 Send Email	Contains an action that triggers on After Record Submit and sends an email that notifies the user's supervisor that the sales order was entered.

The workflow instance executes as follows:

1. The workflow instance initiates when the user creates the sales order record and enters the entry state, State 1 Entry.

   The server trigger on which the record enters the state is Before Record Load.
2. The workflow instance executes actions set to trigger on Entry and marks the field as required. Then, the workflow instance executes the Before Record Load trigger, executes the transition, and the record enters State 2 Field Validation.
3. The server trigger on which the record enters the state is still Before Record Load.

   The workflow instance executes the client trigger when the user edits the required field and makes sure the field has the correct value. The user clicks **Save** and the Before Record Submit trigger executes and the record transitions to State 3 Send Email.
4. The record enters State 3 Send Email on the last trigger that executed, Before User Submit. The workflow instance then executes the next server trigger, which is After Record Submit. The send email action executes and the workflow completes.

Rules and Guidelines for Workflow Triggers and Trigger Execution

Use the following rules and guidelines when working with workflow triggers:

- Make sure actions set to execute on the Entry trigger are valid for the current server trigger.

  If an action executes on the Entry trigger, the action must be able to be executed on the current server trigger for the state. For example, a state contains a Send Email action set to execute on the Entry trigger and the record enters the state on the Before Record Load server trigger. However, the Before Record Load trigger is not a valid trigger type for the Send Email action. Consequently, the Send Email action does not execute.

  You can see which server triggers are valid for an action by viewing the **Trigger On** property for the action. You can also view the help for the action. For a list of all actions, see Workflow Actions Overview.
- Do not add actions to Exit states for any triggers other than the trigger on which the record enters the state.

  Exit states are indicated in the workflow diagrammer. Exit states only execute actions associated with the trigger on which the record entered the state. For example, if a record enters the final state in a workflow on a Before Record Load trigger, actions set to execute on a Before Record Submit or a After Record Submit trigger do not execute. For more information, see Exit States.

- If a user reloads the record when a record is in a state, NetSuite executes the server triggers again, starting with the Before Record Load trigger. Actions set to execute on Entry do not execute again.

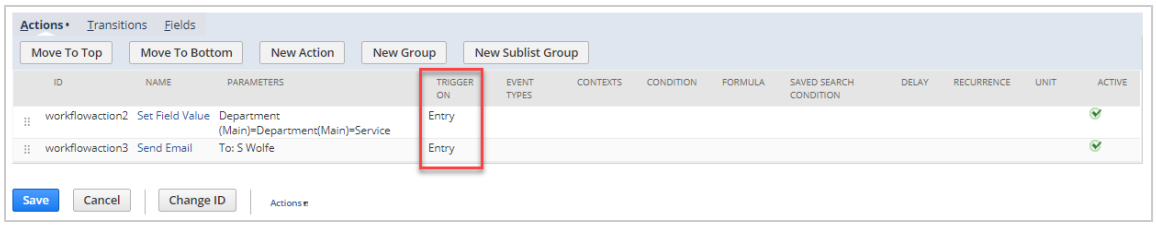
Workflow Actions

Actions are specific tasks performed by a workflow instance based on the properties of a record. You can use actions in workflows to manipulate the fields on the record, add or remove buttons, create new records, send email, and more. After you create a state, you choose the actions you want to execute when the record is in that state. When a record enters a state in a workflow, the workflow instance executes the actions depending on their triggers.

Each action has its own set of parameters, depending on the action type. All actions contain the following elements:

- Action properties. Properties common to most action types. See Action Properties.
- Trigger. The trigger determines when an action executes. See Action Triggers.
- Conditions. The condition determine the additional requirements that must be met for an action to execute. See Action Conditions.

The following screenshot shows two actions that have been added to a state. Both actions are set to execute on the Entry trigger, or when the record in the workflow enters that state. When the record enters the state, the **Department** field is set to **Service** and an email is sent to the supervisor of the sales rep.



Note: For a list of all actions available in SuiteFlow, see Workflow Actions Overview.

More Information about Actions

After you create an action, you can move or copy it to another state, reorder the actions on the **Actions** subtab, inactivate the action, and create action groups.

If the Show Internal IDs preference is enabled on your account, you can see the internal IDs for workflow actions in the following locations:

- Workflow Execution Logs
- Workflow Context Panel
- Workflow Action Group Window

You can use the workflow action IDs to quickly identify every executed action in context and troubleshoot workflows. For more information about the Show Internal IDs preference, see the help topic Setting the Show Internal IDs Preference.

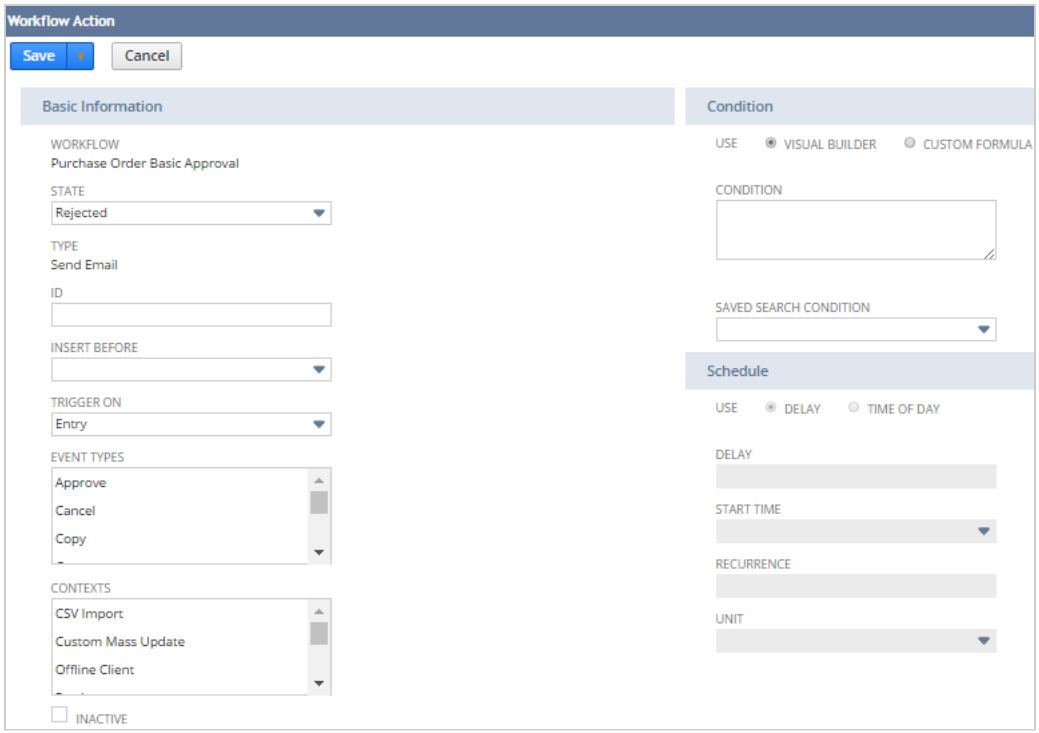
Use the following table to get more information about working with actions in a workflow:

Task	More information
Creating, editing, and deleting actions	Working with Actions
Get more information about action triggers and conditions	Action Triggers and Action Conditions
Get a list of all actions	Workflow Actions Overview
Scheduling an action	Scheduling an Action
Creating action groups	Using Action Groups
Moving and copying actions	Moving an Action
Using actions to accomplish specific tasks	Action Examples, including the following examples: ■ Using Buttons to Execute Transitions ■ Using Buttons for Navigation ■ Executing an Action with a Saved Search Condition ■ Using Conditional Fields with Actions ■ Setting Field Values in Action Definitions

Action Properties

Each action type in SuiteFlow has parameters that are specific to that action type. For example, the Send Email action has parameters that define the sender and recipient. However, most actions contain common properties that you use to define the execution of the action within the state where you create the action.

The following screenshot shows the Send Email action properties:



Note: For more information about each of these properties, see Creating an Action and Scheduling an Action.

Action Triggers

When a workflow instance is triggered, it always executes the actions of the current state and with the appropriate trigger(s), respecting their order. When actions of the current state are executed, the workflow instance considers all transitions from the current state of the appropriate trigger(s), respecting their order. The first transition matching all its conditions is executed. Next, the workflow moves to the target state and continues executing actions of the target state in the same way. The workflow proceeds executing in this manner if possible. When no more relevant transitions exist, the execution session started by the original executing trigger is finished. If the workflow is in a state which has no transitions and the state is marked as Do Not Exit, the entire workflow instance is finished. Otherwise, the workflow is dormant until another trigger reactivates it.

There is no fixed order for the execution of triggers. The workflow definition and events happening on the record (load, view, save, etc.) determine when a trigger is executed.

SuiteFlow triggers are either server side or client side:

- Client side triggers execute in the web client when a user edits the record.

  Client side triggers execute in the following situations:

  - Before User Edit — when the record loads into the browser, before any changes are made to the form.
  - Before Field Edit — before a user completes changes to a field on a record form.
  - After Field Edit — after a user completes changes to a field on a record form.
  - After Field Sourcing — after all dependent field values on a record have been populated.
  - Before User Submit — after a user clicks **Save** on the record, before the Before Record Submit trigger executes.
- Server side triggers execute in the following situations:
  - On Entry — when the record enters the state.
  - On Exit — when the record exits the state.
  - Before Load — before the record is loaded.
  - Before Submit — before the record is saved to the database.
  - After Submit — after the record is saved to the database.
  - Scheduled — when the scheduler executes the workflow based on its schedule definition. For more information, see Scheduling an Action.

Server side triggers can be further classified as follows:

- Record based triggers — execute when something happens on the record — it loads, it is going to be saved, it was saved.
- Workflow based triggers — execute when something happens on the workflow — a state is entered or exited.
- Time based triggers — execute when the scheduled time is reached.

There are a few rules regarding the order or trigger execution:

- When a record is going to be viewed by the user, It executes the Before Load trigger when loading the record.

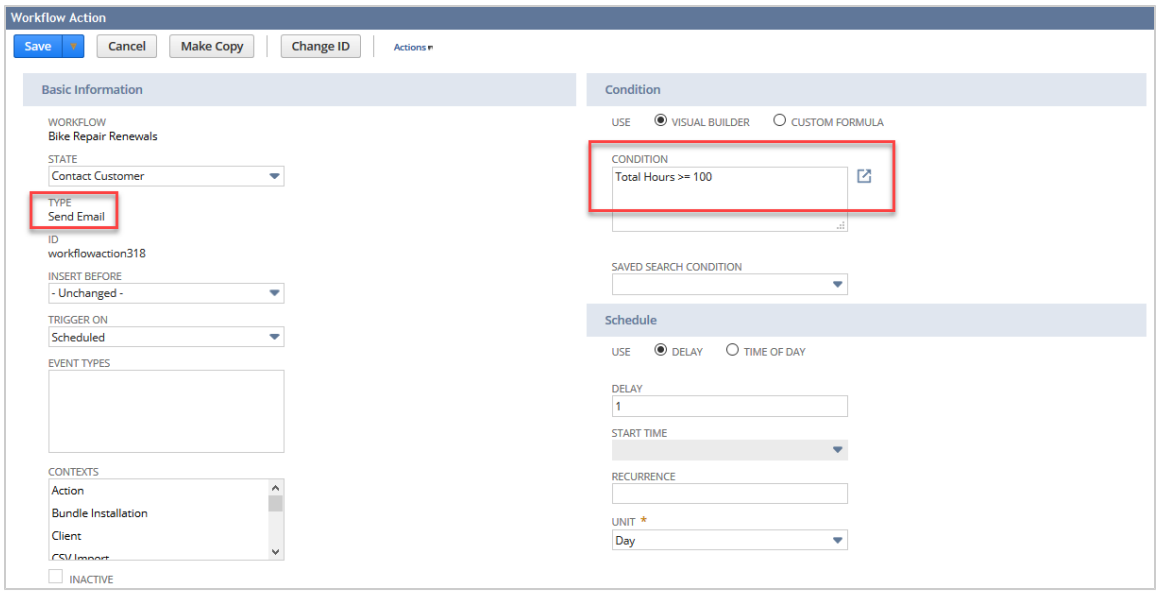
- When a record is going to be edited by the user, it first executes the Before Load trigger when loading the record, then, if the user saves the changes, it executes the Before Submit and After Submit triggers.
- When a workflow moves from state A to state B, it first executes On Exit on state A, then it executes On Entry on trigger state B.
- On Entry triggers do not execute on their own. They execute at the same time as the first ordered server trigger for a state, for example, the On Entry trigger executes when the workflow enters a state simultaneously with the Before Record Load trigger.

Action Conditions

Actions can execute based on a condition. If an action has a condition associated with it, the action executes only if the condition is met. You can create conditions using the Condition Builder or using custom formulas.

If you want to create multiple actions with the same set of conditions, you can use action groups. For more information, see Using Action Groups. For more information about conditions, see Workflow Conditions.

The following screenshot shows a Send Email action that will execute in one day from the time the record enters the state, if the **Total Hours** field on the record is equal to or greater than 100:



Important: If an action executes on a client trigger, a custom formula must be in SuiteScript. If an action executes on a server trigger, a custom formula must be in SQL.

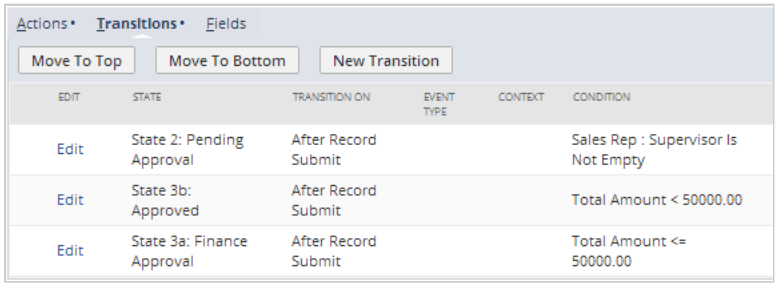
Workflow Transitions

A transition moves the record in the workflow into another state. A state can contain multiple transitions, with each transition having its own condition. A workflow instance executes the first transition that meets the transition condition and the record in the workflow moves to the specified state.

Each transition has the following elements:

- Transition properties. Properties common to all transitions. See Transition Properties.
- Trigger. The trigger determines when a transition executes. See Transition Triggers.
- Conditions. The condition determine the additional prerequisites that must be met for a transition to execute. All conditions must be satisfied to execute the transition. See Transition Conditions.

The following screenshot shows the **Transitions** subtab for a workflow state that contains three transitions. The first condition that evaluates to true executes.



More Information About Transitions

Use the following table to get more information about working with actions in a workflow:

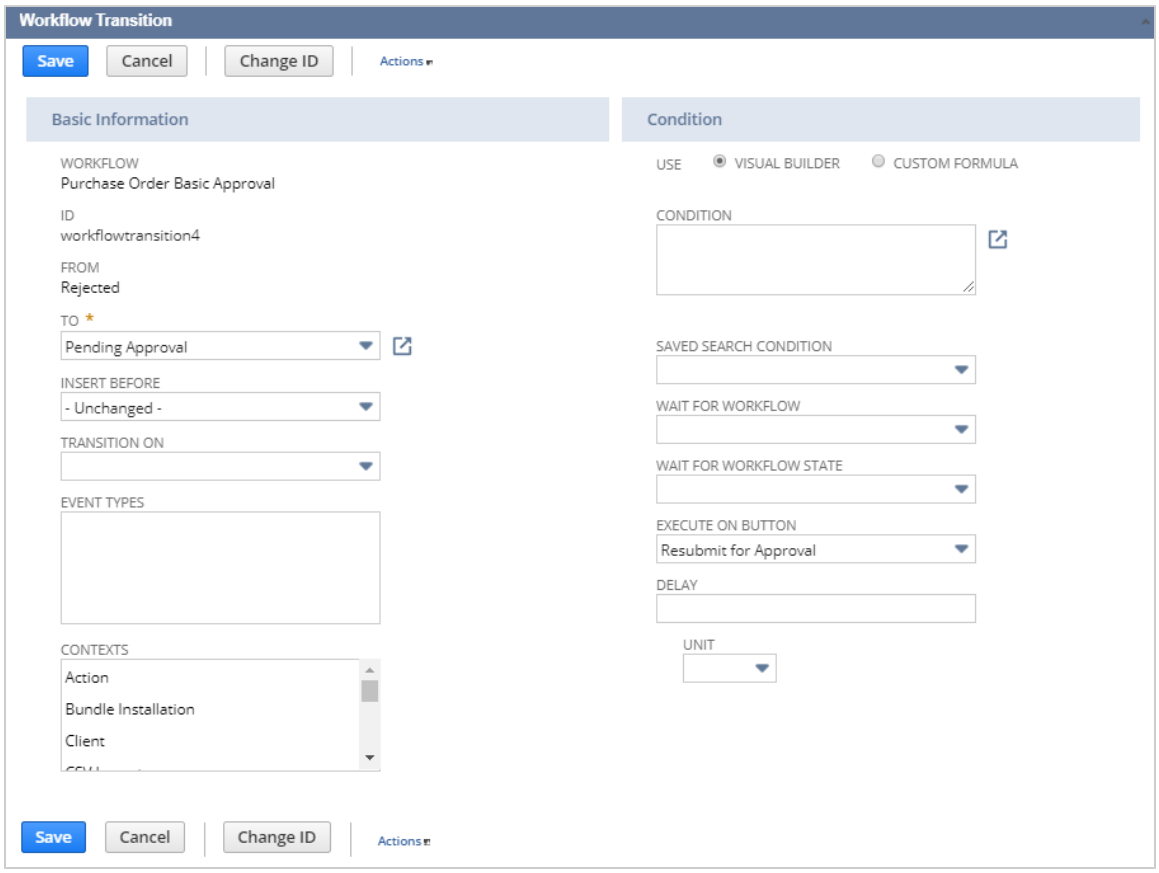
Task	For more information
Creating, editing, and deleting transitions	Working with Transitions
Get more information about transition triggers and conditions	Transition Triggers and Transition Conditions
Scheduling a transition	Scheduling a Transition
Using transitions to accomplish specific tasks	Transitions Examples Includes the following examples: ■ Using Buttons to Execute Transitions ■ Blank Transition Trigger ■ Executing a Transition with a Saved Search Condition ■ Specifying States for Child Workflow Transitions

Transition Properties

Use the transition properties to define when a transition executes. You create a transition in the diagrammer or from an existing state in a workflow.

You can access the properties for an existing transition by clicking a transition in the diagrammer and clicking the **Edit** icon on the **Transition** tab on the context panel.

The following screenshot shows the workflow transition properties:



Transition Triggers

The optional trigger type you select for the **Trigger On** property of a transition determines when the transition executes.

The following table lists the valid transition trigger types:

Trigger Type	Description
Entry	Transition only executes when the record in the workflow first enters the state.
Before Record Load	Transition executes before the record loads into the browser.
Before Record Submit	Transition executes after the user clicks **Save** on the record and before NetSuite writes the record data to the database.
After Record Submit	Transition executes after NetSuite writes the record data to the database.
Scheduled	Transition executes after the record enters the state and meets the defined schedule properties. See Scheduling a Transition.
Blank	Transition executes as soon as the conditions of the transition are met. For more information about leaving the transition trigger blank, see Blank Transition Trigger.

Transition Conditions

You can create different types of transitions, depending on the conditions of the transition. In addition to specifying a condition using the Condition Builder or using custom formulas to define conditions, the **Saved Search**, **Workflow** and **State**, **Button**, and **Delay** and **Unit** properties can also be used to define when a transition executes.

Note: If you define multiple conditions for a transition, all the conditions must evaluate to true for the transition to execute.

The following table describes the transition conditions:

Transition Condition	Description
Condition built with Condition Builder	Use the Condition Builder to create a transition condition. For information about the Condition Builder, see Defining a Condition with the Condition Builder and Defining a Condition with Expressions.
Custom formula	Conditions built using formula values that are evaluated when the workflow instance runs. For more information about custom formulas, see Defining a Condition with Formulas.
Saved search	Use a saved search as a condition for transition execution. The transition executes if the current record in the workflow meets the criteria in the saved search. For more information, see Executing a Transition with a Saved Search Condition.
Transition on button click	Transition to another state based on the click of a button on the current record. You must add the button with the Add Button action before you can select it in this dropdown list. For more information, see Using Buttons to Execute Transitions.
Transition based on another workflow or workflow state	Use the completion of another workflow or another workflow state as a condition for a transition. For more information, see Specifying States for Child Workflow Transitions.
Schedule a transition	Schedule a transition to execute on a delayed period of time after the record enters the state. For more information, see Scheduling a Transition.

Workflow Conditions

You can use conditions on workflow initiation, actions, and transitions. Conditions can be basic field/value comparisons, expressions, or formulas. Use conditions to restrict for whom or when a workflow initiates or actions and transitions execute. You can use user fields in conditions to check the user, role, class, department, location, or subsidiary.

Use one of the following methods to create a condition for a workflow initiation, action, or transition:

- **Condition Builder.** Build conditions or expressions based on fields for the workflow base record or on joined records. See Using the Condition Builder.
- **Formula.** Build conditions by entering formula values or by using the Formula Builder. The formula values are evaluated when the workflow instance runs. See Using a Custom Formula.

More Information About Conditions

Use the following table to get more information about working with conditions in a workflow:

Task	For more information
Creating conditions with the Condition Builder	Defining a Condition with the Condition Builder
Creating conditions with the Formula Builder	Defining a Condition with Formulas
Using conditions with transitions	Transition Conditions
Using conditions with actions	Action Conditions
Examples of using conditions	Condition Examples, including: ■ Referencing Old (Pre-edit) Values in a Workflow ■ Defining Conditions for Customer Credit Hold Field

Using the Condition Builder

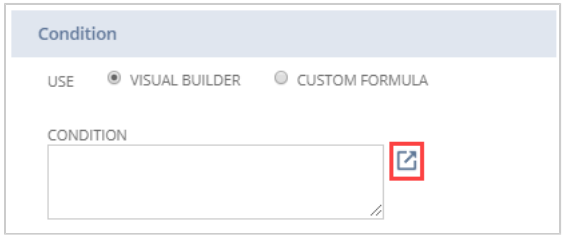
The Workflow Condition Builder lets you specify conditions for workflow initiation, actions, and transitions. You can build conditions based on the fields for the workflow base record or on joined records. You can also use the Condition Builder to create expressions with NOT, AND, and OR operators. You can also use a custom formula. For more information, see Using a Custom Formula.

To create a condition with the Condition Builder, see Defining a Condition with the Condition Builder.

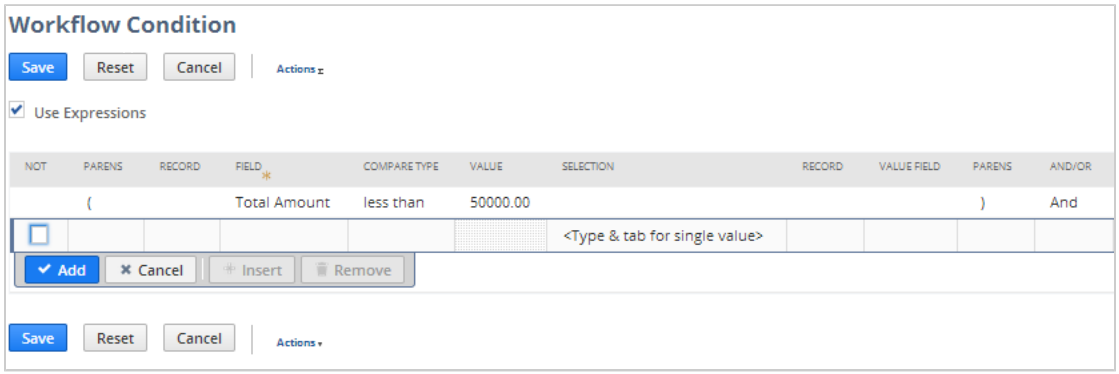
You can access the Condition Builder from the following locations:

- Workflow definition page
- Workflow Action window for specific workflow action in a state
- Workflow Transition window for a state

The following screenshot shows the Condition Builder button on an action:



Click the Condition Builder button to open the **Workflow Condition** window. The following screenshot shows the window with the **Use Expressions** box checked:



Note: You cannot pick multi-select fields in the Condition Builder.

Using a Custom Formula

You can use custom formulas to create conditions for workflow initiation, actions, and transitions. You can build formulas based on the fields for the workflow base record or on joined records. The formula values are calculated dynamically when the workflow instance runs.

Custom Formula Elements

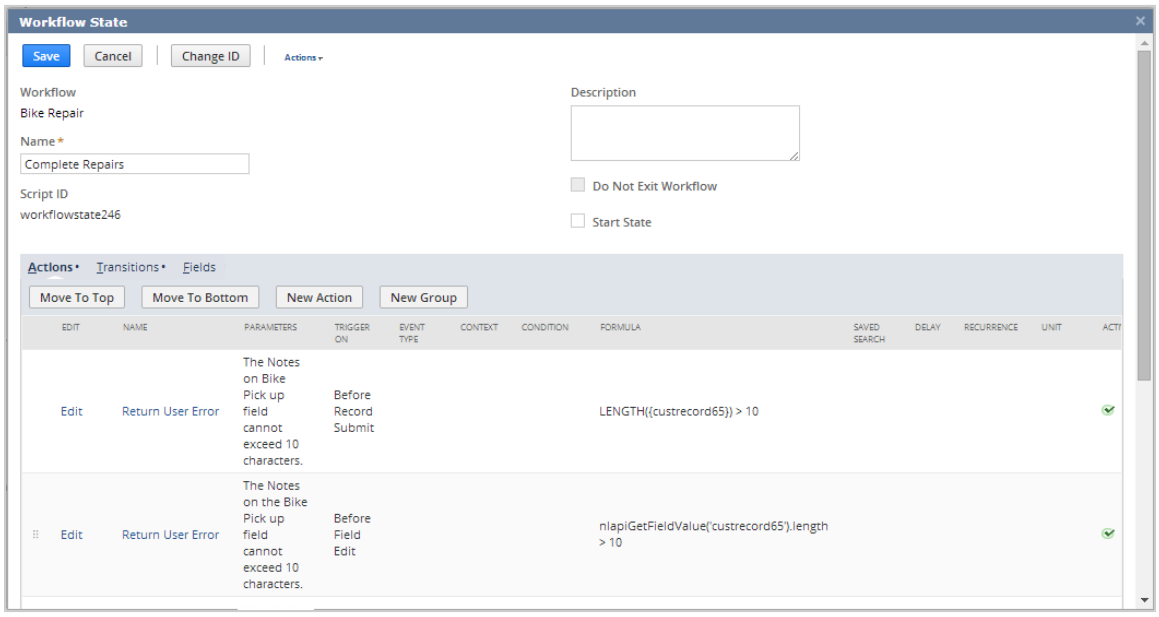
You can use the following elements in a formula:

- **Internal NetSuite field IDs.** Record fields in NetSuite have an internal NetSuite ID. You can reference fields with this internal ID. For more information about working with formula fields, see the help topic Creating Formula Fields.
- **SQL functions.** You can use SQL functions in a condition for workflow initiation or in a condition on a transition or action that executes on a server trigger. For more information about SQL functions, see the help topic SQL Expressions.

  For example, you can use the following condition to restrict the custrecord65 field to 10 characters or less in a Return User Error action: `LENGTH({custrecord65}) > 10`.
- **JavaScript/Suitescript API functions.** You can only use JavaScript or SuiteScript in a condition on an action that executes on a client trigger.

You can enter formulas directly in the **Formula** box or you can use the Formula Builder. See Using the Formula Builder.

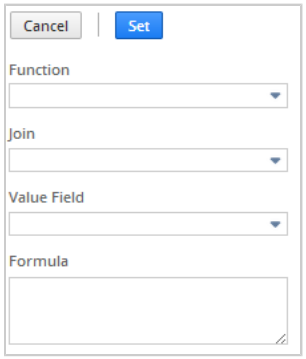
The following screenshot shows two actions with custom formulas for conditions, where the SQL formula occurs on a server trigger and the SuiteScript API formula occurs on a client trigger:



Using the Formula Builder

Similar to the Condition Builder, you can use the Formula Builder to create custom formulas. To use the Formula Builder, select the SQL functions or JavaScript functions and the fields, and the builder creates the formula in the **Formula** box. To create a formula with the Formula Builder, see Defining a Condition with Formulas.

The following screenshot shows the Formula Builder for a SQL expression:



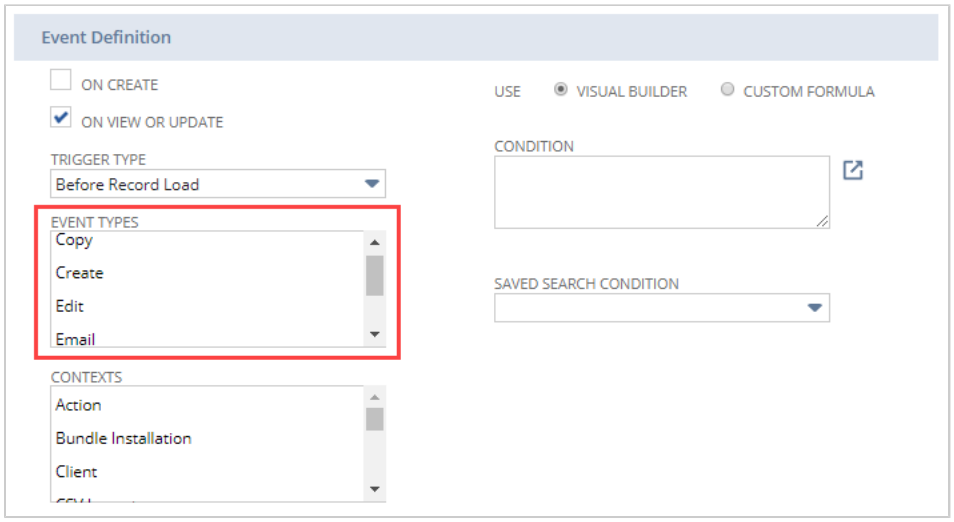
Workflow Event Types

You can define workflows to initiate or execute actions and transitions on specific event types. You can direct workflows to initiate when records of the base record type are created, updated, or viewed with the **On Create** and **On Update** event types.

SuiteFlow includes additional event types with the **Event Types** field on the workflow definition, action, and transition definition pages. These events represent the user interface activity used to create, view, or update the record for a workflow.

Use these event types to further limit when workflows initiate or when actions and transitions execute. For a full list of all event types, see Event Types Reference.

The following screenshot shows the **Event Types** field on the workflow definition page:



The Workflow Manager dynamically populates the event types you can select in the **Event Types** field. The following table describes the criteria for including an event type in the field:

Criteria	Description
Server trigger	The Workflow Manager displays different event types based on the server trigger, which includes Entry and Exit for actions and transitions in a state. For example, the Before Record Load trigger type and the **View** event type are compatible. The only time you can view a record is when it loads into the browser. However, if you set the workflow to initiate on the After Record Submit trigger, the **View** event type is not available as an event type.

Criteria	Description
	See Event Types Reference for information about the relationship between server triggers and event types.
Base record type	Some events are specific to a record type. For example, the Mark Complete event type will appear in the **Event Types** field only if the workflow base record type is Task or Phone Call. Other event types may appear depending on the features enabled in the account. For example, you must have the Pick, Pack, and Ship feature enabled to see Pack and Ship event types on an Item record such as Item Fulfillment.

Rules and Guidelines for Event Types

Use the following general rules and guidelines with event types:

- The **Event Types** field is not available for actions that execute on a client trigger or the Scheduled trigger for workflow initiation or action or transition execution.
- The Workflow Manager does not limit all incompatible relationships between **On Create** and **On Update** and the corresponding events in the **Event Types** field.

  For example, you can use the View event type for workflow initiation. However, the view event only occurs if the record opens in view (read-only) mode, rather than in edit mode. Consequently, if you define the workflow to only initiate on the creation of a record and select the View event type, the view event never occurs and the workflow never initiates.

More Information About Event Types

Use the following table to get more information about working with event types in a workflow:

Task	For more information
Specify the event type in a workflow definition	Creating a Workflow
Specify the event type in an action or transition	Creating an Action and Creating a Transition
Get a list of all event types and when they are available	Event Types Reference
Get examples of using the Print, Mark Complete, and Drop Ship event types	Examples of Event Types

Execution Contexts and Workflows

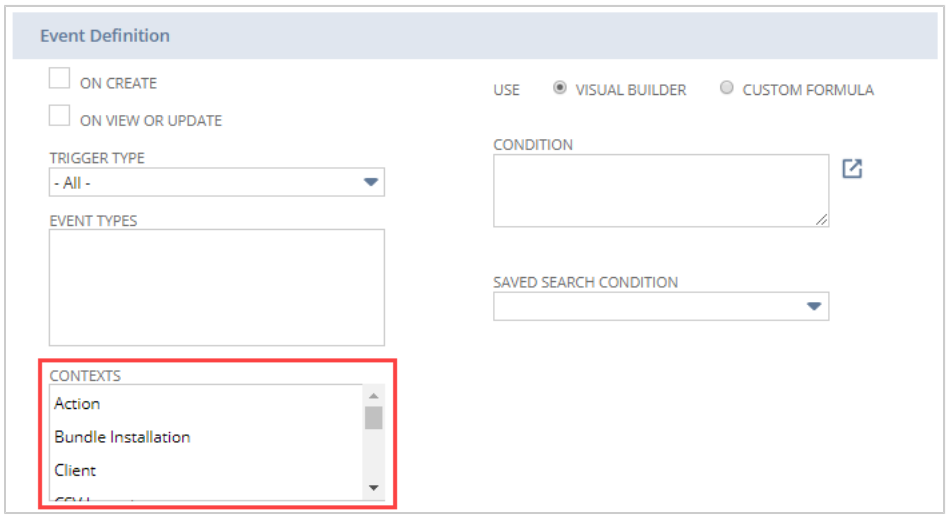
Execution contexts provide information about how or when a SuiteScript script or SuiteFlow workflow is triggered to execute. For example, a script can be triggered in response to an action in the NetSuite application, or an action occurring in another context, such as a web services integration. You can use execution context filtering to ensure that your scripts or workflows are triggered only when necessary.

You can specify that a script or workflow should execute only in certain contexts, and this filtering can improve performance in contexts where the script or workflow is not required. You can also determine the execution context for a running script programmatically and use different logic depending on the context. For more information about execution contexts, see the help topic Execution Contexts.

There are many execution contexts available in NetSuite. For a full list of execution contexts, see the help topic Execution Context Types.

Set the context for workflow initiation, actions, and transitions using the **Contexts** field on the workflow, action, or transition definition pages.

The following screenshot shows the **Contexts** field on the workflow definition page. This workflow does not initiate unless a sales order is created on the Web Store:



More Information About Execution Contexts

Use the following table to get more information about working with context types in a workflow:

Task	For more information
Specify the context type in a workflow definition	Creating a Workflow
Specify the context type in an action or transition	Creating an Action and Creating a Transition
Get a list and description of all context types	Execution Context Types
Get examples of using the CSV Import, User Event Script, and Custom Mass Update context types	Context Type Examples

Workflow Custom Fields

Important: Workflow Custom Fields and Workflow State Custom Fields cannot be successfully used in client-side actions and conditions.

Custom fields are variables that you can use in workflows. Use custom fields to collect and track data for a workflow instance or a state in a workflow. Each type of custom field is defined by the scope of the field within the workflow.

SuiteFlow includes the following custom field types:

Custom Field Type	Application
Workflow Field	Workflow fields are associated with one particular workflow definition and can store a unique value for each instance of the workflow. These fields are available to all states within a workflow, in workflow conditions, and in actions such as Set Field Value. Workflow fields appear on the **Fields** tab of the workflow definition page or on the **Fields** view for a workflow in the context panel.

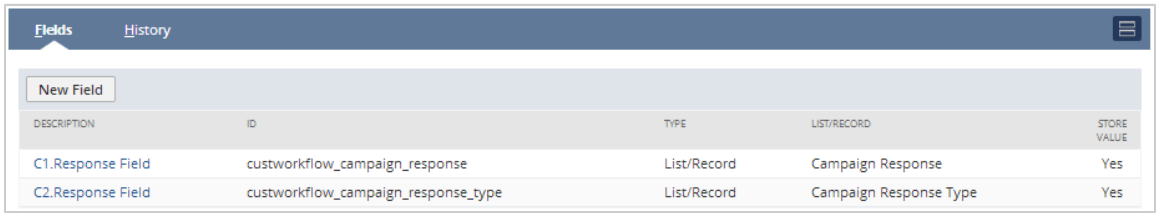
Custom Field Type	Application
	You can store the value of the field in the database or store the value until the workflow instance completes. If you do not check the **Store Value** box, the database does not save the field value and the value will only be accessible in the current workflow instance. Use workflow fields in the following situations: ■ Track records created by the by the Create Record action. Assign the new record to an workflow field and subscribe to the record, using the workflow field, with the Subscribe To Record action. See Create Record Action. ■ Store a return value from a custom action script in a workflow field. See Storing a Return Value from a Custom Action Script in a Workflow Field. For more information about creating and using a workflow field, see Creating and Using Workflow Fields.
State Field	State fields function the same as workflow fields. However, workflow fields apply to the workflow instance and state fields are limited to the state where you create it. State fields appear on the **Fields** subtab of the **Workflow State** window or on the **Fields** view for a state in the context panel. You can store the value of the field in the database or store the value until the workflow instance completes. For more information about creating and using a state field, see Creating and Using State Fields.

Note: You can also create workflow definition fields that apply to all workflows in a NetSuite account. To create a workflow definition field, use SuiteBuilder to create an **Other Record Field** of type **Workflow**. After you create the field, the field appears on the workflow definition page for all workflows in the account. For more information, see the help topic Creating a Custom Field.

Use these fields to collect or track data across all workflows in an account. You can also use the field as search criteria when you perform a workflow search.

Sample Workflow Field

The following screenshot shows workflow fields used to track campaign responses in a lead nurturing workflow:



Workflow Revisions

Important: Workflows created prior to 2017.2 are listed as revision number 0. Subsequent changes made to existing workflows increase the revision number.

Workflow revisions let you see that changes have been made to a workflow.

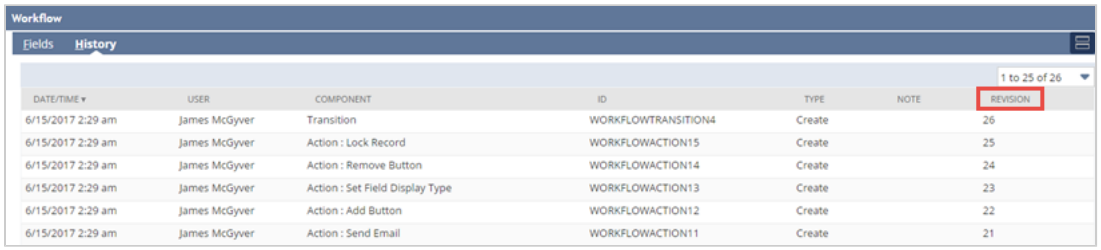
Workflow revisions begin after a workflow is created. When a workflow is created, its initial revision number is 1. The addition of a state is considered to be revision 2. The two simultaneous creation events cause the workflow revision to be listed as 2. Subsequent changes increase the revision number by 1. For example, adding a custom workflow field or adding an action would each increase the revision number by 1. Any change made to a workflow's definition or any of its children also increase its revision number.

For more information about tracking workflow revisions, refer to the following sections:

- Workflow Revisions on the History Subtab
- Workflow Revisions on the Workflows List Page
- Workflow Revisions in Searches

Workflow Revisions on the History Subtab

Workflow revisions can be tracked on the Workflow Definition Page History subtab in the Revision column.

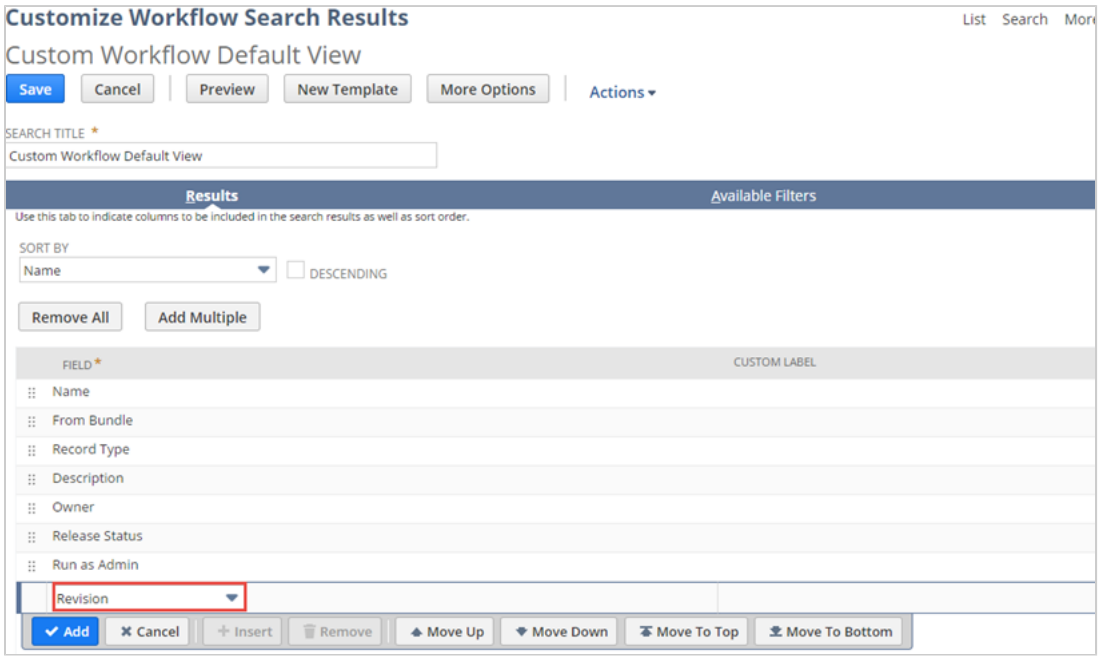


The following changes increase the workflow revision and are listed on the Workflow Definition Page History subtab:

- Creating or changing a workflow definition.
- Creating, changing or deleting a workflow state.
- Moving a state on the Workflow Diagrammer.
- Creating, changing or deleting a workflow transition.
- Creating, changing or deleting any workflow actions or action groups.
- Creating, changing or deleting Workflow Custom fields and Workflow Custom State fields.
- Creating, changing or deleting a workflow condition.
- Changing the script id of any workflow component increases the revision and sets the note to "{1} changed to: {2}" where {1} is the old id and {2} is replaced with the new id.
- Bundle updates or installs increase the revision. The entry to history remains the same: "Workflow has been modified by bundle {1}" where {1} is replaced by the bundle id.
- Changing the release status or logging on the workflow definition of a workflow locked by a bundle adds the note of what changed.
- From the Workflows list page, setting the workflow to inactive or active increases the revision and logs a note "set to {1}" where {1} is replaced with the new value.

Workflow Revisions on the Workflows List Page

You can view a workflow's revision number on the Workflows list page by customizing the page to include the new **Revision** column.

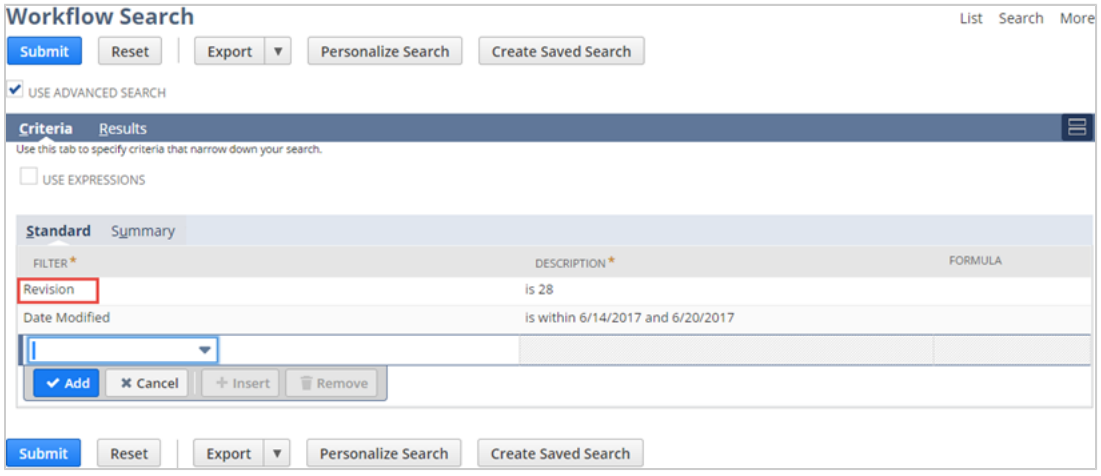


To add the Revision column to the Workflows list page:

1. Go to Customization > Workflow > Workflows.
2. Click **Edit View**.
3. From the **Field** list, select **Revision**, then click **Add**.
4. Click **Save**.

Workflow Revisions in Searches

You can search for workflow revisions by creating an advanced workflow definition search that includes the Revision column as a Standard Filter. For details, see Workflow Definition Search.



Workflow Initiation

For all workflows, you must specify when NetSuite initiates an instance of the workflow. You can set a workflow to initiate based on an event or based on a schedule.

If a workflow initiates based on an event, the workflow runs when a record of the base record type is created or updated. If a workflow initiates on a schedule, the workflow runs at the date and time in the schedule, against a set of records that meet criteria defined in a saved search.

A workflow must include at least one state before it initiates. Whenever a workflow initiates, NetSuite creates a new instance of the workflow. Multiple instances of the same workflow can be running simultaneously on different records.

You specify when the workflow initiates on the workflow definition page:

- **Based on an event.** The workflow runs when a user creates or updates a record of the base record type for the workflow, against each record that is created or updated. For example, a lead nurturing workflow initiates when a Sales Rep creates a Lead record. This workflow runs against that newly created record.

  For more information, see Initiating a Workflow on an Event.
- **Based on a schedule.** The workflow runs on a recurring basis at defined time intervals, or on a one-time basis at a specific day and time. The workflow runs against records that are returned as results of a selected saved search. For example, you can create a workflow that runs every month and uses a saved search to find customers with at least one sales order, and sends the customer a welcome email.

  For more information, see Initiating a Workflow on a Schedule. To test a scheduled workflow, you can execute it on demand.

Typically, you specify workflow initiation on the workflow definition page. However, you can also initiate a workflow in one of the following ways:

- Using SuiteScript. Use SuiteScript to programmatically initiate a workflow on-demand against specific records. You must first create the workflow on the workflow definition page and then use SuiteScript to initiate the workflow. For more information, see Using SuiteScript to Initiate a Workflow.
- Using the Initiate Workflow action. Run another workflow from within the current workflow. For more information, see Initiate Workflow Action.

Initiating a Workflow on an Event

To initiate a workflow instance on an event, select **Event Based** under **Initiation** on the workflow definition page and then select one or both of the following events for the workflow base record type:

- **On Create.** Initiate an instance of the workflow when the record is created.
- **On Update.** Initiate an instance of the workflow when the record is viewed or updated.

You must also select the server trigger type for workflow initiation. See Workflow Initiation Triggers.

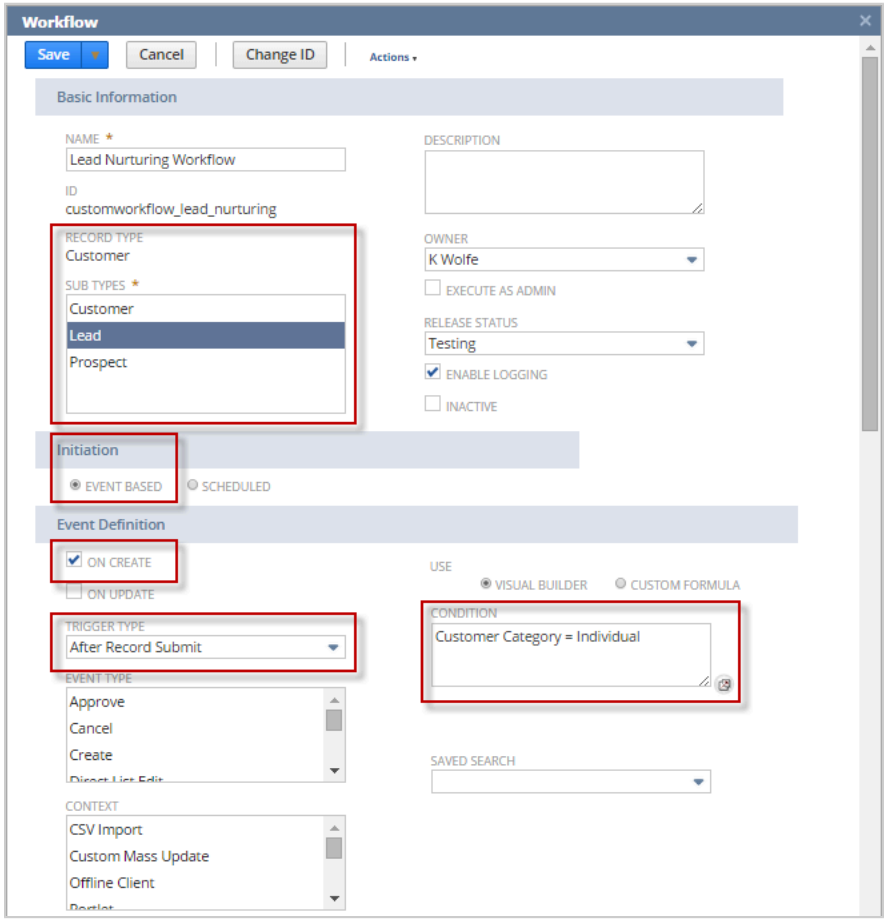
The following table describes optional requirements that you can use to further define when a workflow instance initiates:

Option	Description
Event Types	User interface activity used to create, view, or update the record. The trigger type determines which event types are available. See Workflow Event Types.

Option	Description
Contexts	NetSuite functionality or feature used to create, view, or update the record. See Execution Contexts and Workflows.
Condition	Use the Condition Builder or Formula Builder to define requirements that must be met for a workflow instance to initiate. See Workflow Conditions.
Saved Search	Initiate the workflow if the record for the workflow meets the criteria in the saved search. The saved search must run on the same record type as the base record type for the workflow. The saved search must also include at least one filter set on the **Criteria** subtab for the saved search.

For example, you can create a lead nurturing workflow to run when a record is created and saved. Select **On Create** for the event based initiation and select the **After Record Submit** trigger. You can also create a condition so the workflow instance only runs on lead records with customer category of **Individual**.

The following screenshot shows the lead nurturing workflow:



Workflow Initiation Triggers

For a workflow to initiate on a create or update event on a record, you must also set the initiation trigger type in the **Trigger Type** dropdown list.

The following table describes when the workflow initiates for each trigger type:

Trigger Type	Description
- All -	The workflow initiates for any triggering event. For example, the workflow initiates when a record is viewed because the Before Record Load trigger executes when the record is viewed. Best practice is to choose a more specific trigger type for workflow initiation because the workflow initiates for any action on the record with this trigger.
Before Record Load	The workflow initiates and the record goes into the entry state when you load a record by clicking **New** or **Edit**, or when you view the record.
Before Record Submit	The workflow initiates and the record goes into the entry state after you click **Save** on the record and before NetSuite writes the record data to the database.
After Record Submit	The workflow initiates and the record goes into the entry state after the NetSuite writes the record data to the database.

Note: For more information, see Server Triggers and SuiteFlow Trigger Execution Model.

Initiating a Workflow on a Schedule

When you configure a workflow to run on a schedule, the workflow executes on the results of a saved search. You can schedule the workflow to run one time, run every 30 minutes, or run at a certain time on a daily, weekly, monthly, or yearly basis. When you choose to initiate workflows on a scheduled basis, you select a saved search and then define the schedule on which the workflow runs. For more information, see Scheduling a Workflow.

Before you define the schedule for the workflow, you must create the saved search. The saved search must run on the same record type as the base record type for the workflow. The saved search must also include at least one filter set on the **Criteria** subtab for the saved search. On the workflow definition page, select the **Scheduled** option, select the saved search, and set the schedule for the workflow.

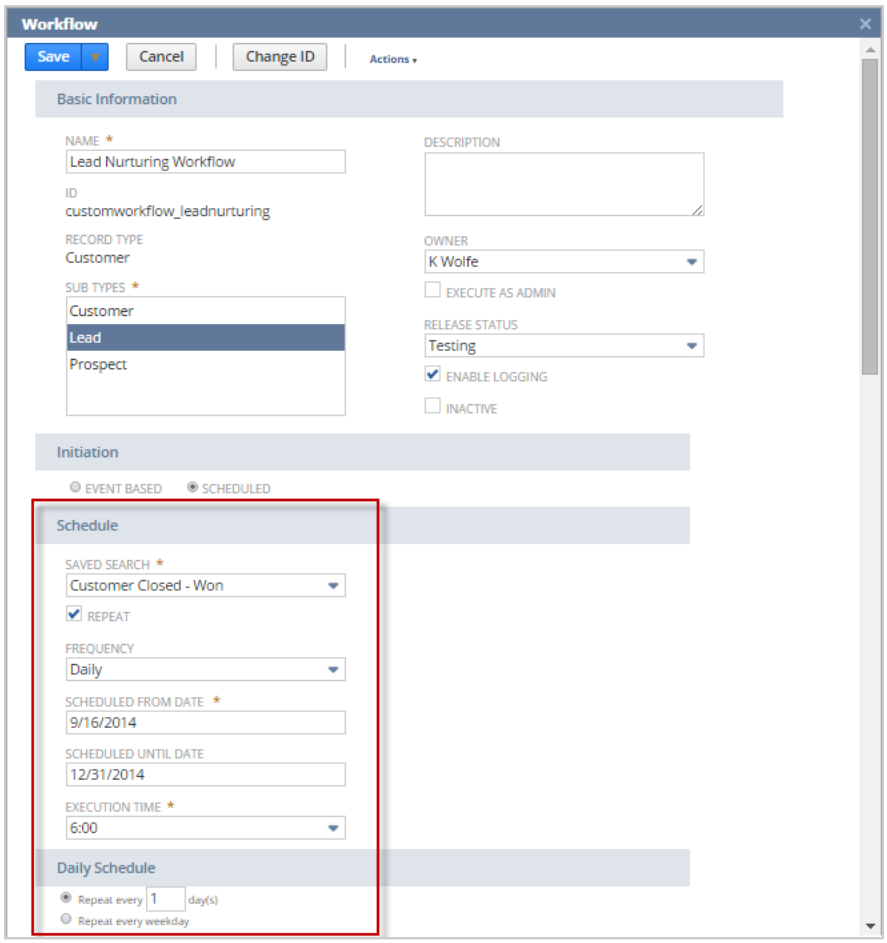
At the scheduled time, NetSuite runs the saved search and initiates a workflow instance to run on each of the records in the search results. For example, if the saved search returns 10 records, NetSuite initiates 10 instances of the workflow. Scheduled workflows always run on all of the records in the search results, not only the records you can see on the main page (for summary type result) of the saved search.

NetSuite runs multiple workflows in parallel. Running in parallel benefits accounts that have multiple scheduled workflows that operate on large saved search result sets. Each workflow instance runs as an administrator and any system note displays “System” as the user name.

There is no inherent order in which scheduled workflows start and finish. For example, three scheduled workflows, A, B, and C, are all scheduled at the same time. NetSuite simultaneously runs each saved search and initiates workflows on the results. Depending on the workflow and the number of search results, any of the workflows may complete before the other two.

Note: Only scheduled workflows with a release status of Released execute on a schedule. Also, you cannot set a condition for the scheduled workflow initiation. All workflow initiation criteria must be specified when you create the saved search.

The following screenshot shows a lead nurturing workflow scheduled to run against Customer Lead records on a daily basis:

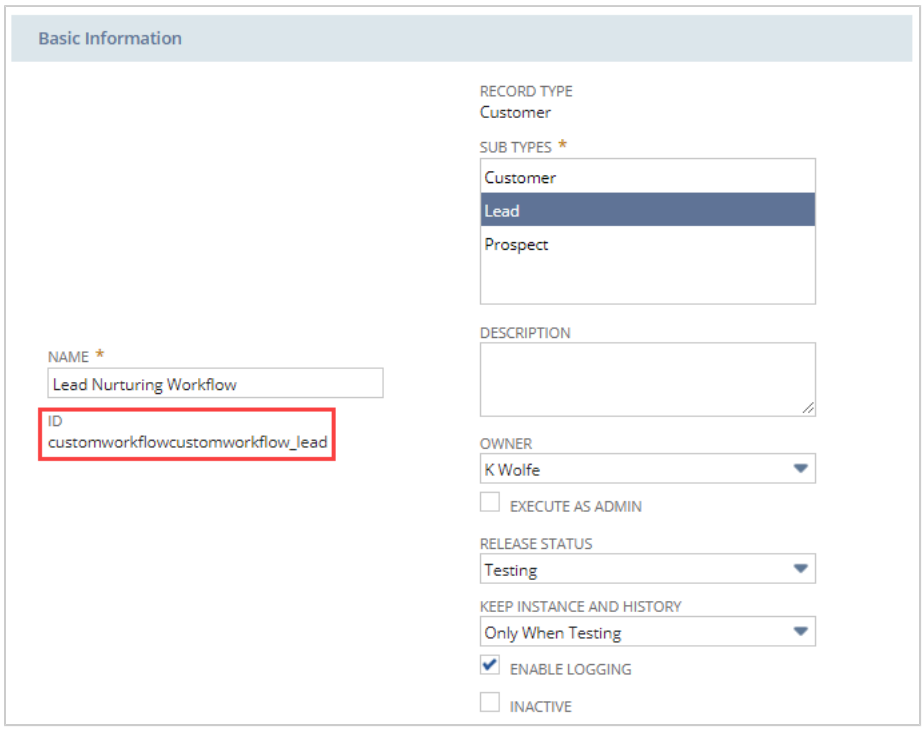


For an example of how to create a saved search and use it in a scheduled workflow, see Welcome Email Sent to Customers Three Days After First Order Workflow.

Using SuiteScript to Initiate a Workflow

You can use a SuiteScript 2.x workflow action script to programmatically initiate a workflow on demand. Use the numeric workflow internal ID or custom script ID, the script ID of the base or subrecord types, and the internal ID of the record to identify the workflow and the record on which the workflow initiates. The workflow custom script ID is specified in the ID field of the workflow definition page. For more information about workflow action scripts, see the help topic Creating and Using Workflow Action Scripts.

The following screenshot shows the script ID location:



Dynamic Mode for Workflows

Workflows almost always run in dynamic mode. Consequently, the dynamic effects of setting field values including validation, sourcing, and calculation are visible in real-time when running workflows. This visibility lets subsequent actions and transitions to react to these dynamic effects.

See the following sections:

- Dynamic Versus Standard Mode in Workflows
- Order of Setting Fields in Standard and Dynamic Modes
- Dynamic Mode Limitations in Workflows
- Record Modes and User Event Scripts
- Record Module Method Considerations

Dynamic Versus Standard Mode in Workflows

The following example shows the difference between dynamic and standard mode in a workflow.

A workflow on a Task record* has two Set Field Value actions in the following order:

1. Set the **Status** field to **Completed**.
2. Set the **Message** field to **Completed after due date!** when the **Date Completed** field is set to a date after the specified due date.

*The Task record has a built-in logic which automatically sets the **Date Completed** field to today when the **Status** field is changed to **Completed**.

In dynamic mode, the **Date Completed** field is automatically set to today's date when the Status field is set to **Completed**. If today's date is later than the task's specified due date, the **Message** field is set to **Completed after due date!** This is possible in dynamic mode because the second action can immediately see the first action's effects.

In the same case in standard mode, the **Message** field remains empty, because the second action cannot see the first action's effects. The record must first be saved for the effects of the first action to be seen on the record.

Order of Setting Fields in Standard and Dynamic Modes

In standard mode, regardless of the order of Set Field Value actions, values are effectively set and the dynamic effects are executed in a pre-defined order when the record is saved after the workflow runs.

In dynamic mode, the order of the Set Field Value actions needs to match the order that a user interacts with the record on the UI. This is because the actions' related dynamic effects are immediately executed in this order within the workflow's execution.

The following example shows problems with ordering Set Field Value actions.

A workflow has two Set Field Value actions in the following order:

1. **Contact** field = **John Black from Big Company**
2. **Company** field = **Big Company**

In standard mode, the workflow actions run without error. First, the **Company** field is set to **Big Company**, then the **Contact** field is set to **John Black from Big Company**.

In dynamic mode, the workflow would fail, resulting in an error similar to **Invalid contact reference key for company 165**. This is because the contact is set, but the company has not yet been set. Therefore, John Black from Big Company is not considered to be a valid value for the **Contact** field and an error results.

Dynamic Mode Limitations in Workflows

In some cases, SuiteFlow cannot use dynamic mode. These cases include actions and transitions on Before Record Load.

Workflow Audience

There are two types of workflow users:

- Users who can create and view workflows
- Users who can run workflows

Users Who Can Create and View Workflows

NetSuite administrators and users with the Workflow permission assigned to their role can create and view workflows. Administrators can modify any workflow in the system, even if they are not the owners of a workflow. Users who are not an administrator, but who have the Workflow permission, can create and edit any workflows on records for which they have the full permission level assigned. To access, create, or

modify a workflow, it is also necessary for users to have the feature related to the workflow base record enabled in their account.

Administrators can assign the Workflow permission to a role by going to Setup > Users/Roles > Manage Roles. See the help topic Customizing or Creating NetSuite Roles.

Administrators and users with the Workflow permission on their role can also perform the following tasks:

- View existing workflows. View existing workflows at Customization > Workflow > Workflows. See Viewing Existing Workflows.
- Perform workflow administration. Execute a saved search for workflows or workflow instances, cancel workflows, perform mass updates, and bundle workflows. See Workflow Administration.
- Specify the workflow audience. For more information, see Defining the Workflow Audience.

Users Who Can Run Workflows

Workflow instances run on records transparently to the user. Users create or update a record and do not know that by creating, viewing, or updating a record they are putting the record into a workflow.

Workflows run for any user depending on the following conditions:

- **User has access to the base record**. If your workflow requires that users have view, edit, and create permission levels for the base record, you should check that the roles of all intended users have the appropriate permission and access levels. Workflows do not run for users who do not have the required permission levels. See Workflow Base Record.

  For example, most users do not have view or create access to Employee records. If you create a workflow to run on Employee records, this workflow does not run for users who do not have access to the Employee record type.
- **User has access to any joined records**. If your workflow sets a field on a record, based on the value of a field in a related record, users must also have at least the View permission level for the related record type. Otherwise, the workflow does not work as intended.

  For example, a workflow that uses a Set Field Value action to set a value on a Purchase Order from a related Vendor or Employee record, then all users accessing the Purchase Order also need at least the View permission level for Vendor and Employee record types.

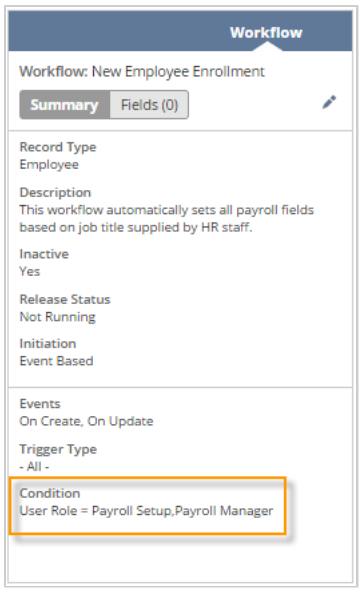
  For more information about a possible workaround for issues with joined records, see Execute As Admin.
- **Workflow is released.** The release status of workflows can be set to Suspended, Not Initiating, Testing, or Released. For more information about the release status, see Release Status.

Defining the Workflow Audience

The audience for any workflow is determined by those users who have view, edit, or create level access to the base record type for the workflow. However, you can use conditions on workflow initiation, actions, and transitions to further refine the scope of a workflow.

Use a condition on the workflow, action, and transition definition pages to define conditions that determine the audience of the workflow. You can use the fields available on the base record type as well as any fields available to related records to define the users for whom a workflow initiates or executes actions.

For example, you can use the User Role field to limit workflow initiation for users logged in with the Payroll Setup or Payroll Manager roles:


Workflow
Workflow: New Employee Enrollment
Summary
Fields (0)
Record Type
Employee
Description
This workflow automatically sets all payroll fields based on job title supplied by HR staff.
Inactive
Yes
Release Status
Not Running
Initiation
Event Based
Events
On Create, On Update
Trigger Type
- All -
Condition
User Role = Payroll Setup,Payroll Manager

Creating Your First Workflow

This tutorial describes the necessary steps to create and run a workflow in NetSuite. Use the steps in the tutorial to become familiar with the Workflow Manager user interface and the elements of a workflow.

In this tutorial, you create a workflow that runs when a user creates a new Opportunity record. The workflow makes the **Title** field on the opportunity record required and sends the user to the Opportunities list page after they save the opportunity.

To view, create, and edit workflows, you must have the appropriate permission and permission level required for working with the base record types in the workflow. For access to all SuiteFlow functionality, use the Administrator role. For more information, see Required Permissions for SuiteFlow.

The following table describes the tutorial steps. Each step builds on the previous step, so you must complete them in order.

Step	Description
Define basic workflow properties	Create the workflow and define the basic properties on the workflow definition page, including the workflow name, workflow base record, owner, and release status. See Step 1 Define Workflow Basic Information.
Define workflow initiation	Define the workflow initiation on the workflow definition page. The workflow initiates on creation of an Opportunity record. See Step 2 Define Workflow Initiation.
Define the workflow condition	Define a condition that must be met for the workflow to initiate. See Step 3 Define the Workflow Condition.
Create workflow states	Create the two states required by the workflow: the entry state and the exit state. See Step 4 Create Workflow States.
Create actions	Create the actions to make a field required and to go to the Opportunities list page. See Step 5 Create Actions.
Create a transition	Create a transition between the entry and exit states. See Step 6 Create a Transition
Initiate and validate the workflow	Create an opportunity and verify that the workflow executes as expected. See Step 7 Initiate and Validate the Workflow.

Tip: See the **Related Topics** sections in each step to get more information about the concepts or properties.

Step 1 Define Workflow Basic Information

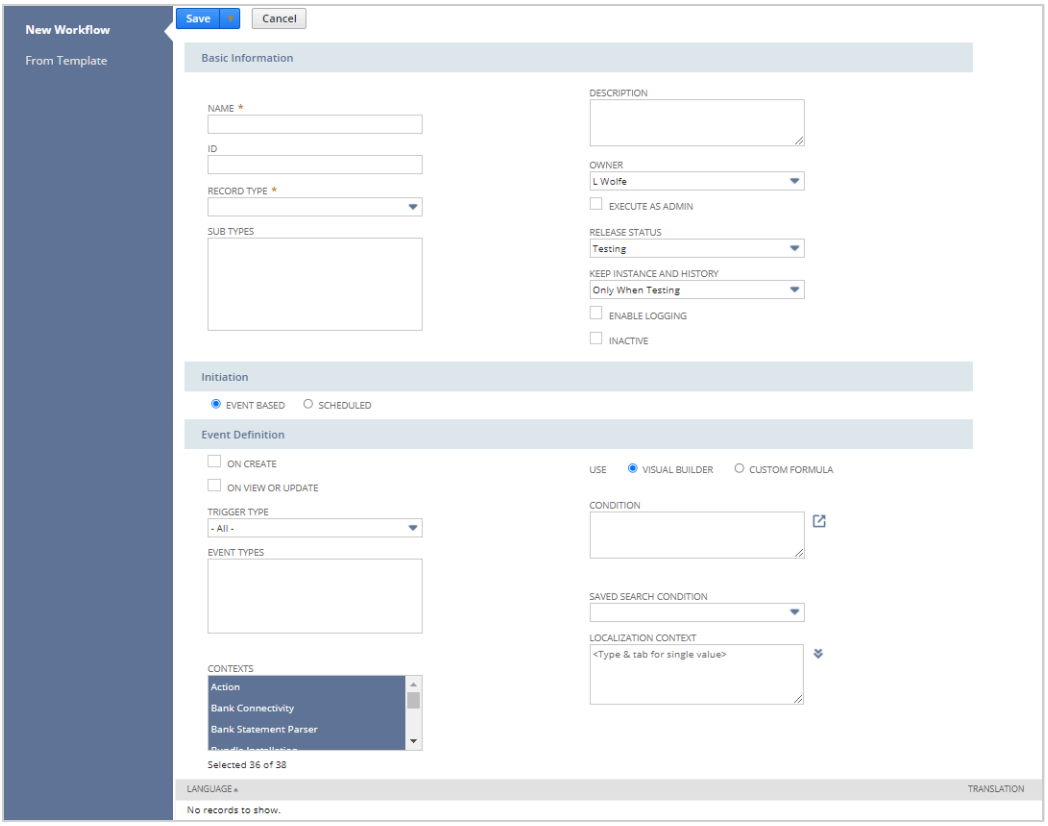
This step shows how to use the Workflow Manager to create a workflow and define the workflow properties, including workflow name, workflow base record, owner, and release status, on the workflow definition page.

To create a workflow and define the basic properties:

1. To create a new workflow, go to Customization > Workflow > Workflows > New. The workflow definition page opens with the following default settings:
   - **Owner**: Should be your name

- **Release Status**: **Testing**
- **Keep Instance and History**: **Only When Testing**
- **Event Based**: Selected
- **Visual Builder**: Selected
- **Trigger Type**: **All**
- **Contexts:** All selected

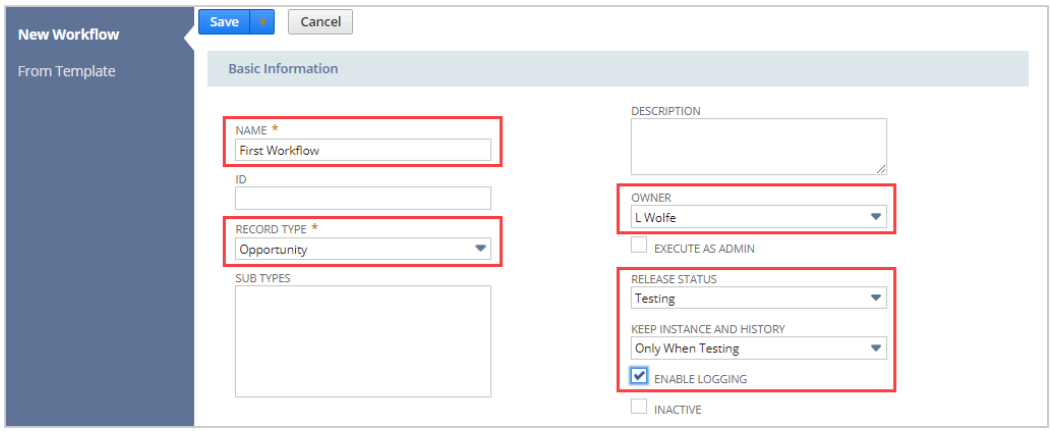
The following screenshot shows the workflow definition page and the default settings:



2. Under **Basic Information**, enter the following properties:

Property	Value
Name	First Workflow
Record Type	Opportunity
Owner	Verify that your name appears
Release Status	Testing
Keep Instance and History	Only When Testing
Enable Logging	Checked

Verify that the workflow properties look like the following screenshot:



3. To continue with the tutorial, continue to Step 2 Define Workflow Initiation.

Note: Record sub types are not available for an Opportunity record and the **Sub Types** field is only available for Item, Transaction, and Customer record types. For more information about record sub types, see Workflow Base Record.

Step 2 Define Workflow Initiation

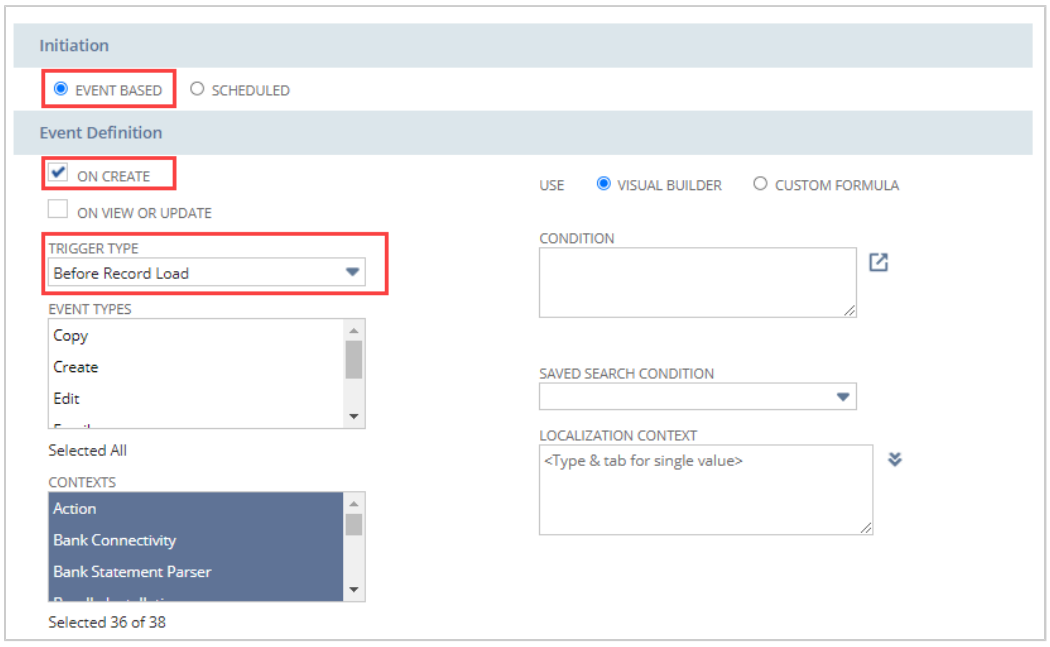
This step shows how to use the Workflow Manager to specify for the instance of a workflow to initiate on an event, when a record is created.

To define the workflow initiation:

1. If you have not already done so, complete Step 1 Define Workflow Basic Information.
2. On the workflow definition page, select the following properties:

Property	Value
Event Based	Selected
On Create	Checked
Trigger Type	Before Record Load

Verify that the workflow properties look like the following screenshot:



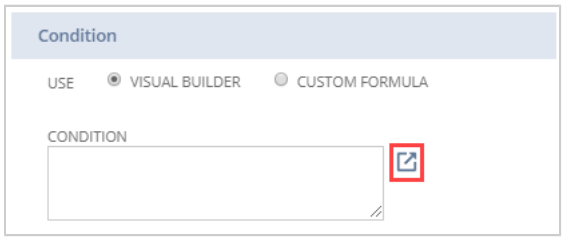
3. To continue with the tutorial, continue to Step 3 Define the Workflow Condition.

Step 3 Define the Workflow Condition

This step shows how to use the Condition Builder in the Workflow Manager to define a condition for a workflow initiation. An instance of the workflow only initiates for records that meet the condition.

To define a condition for workflow initiation:

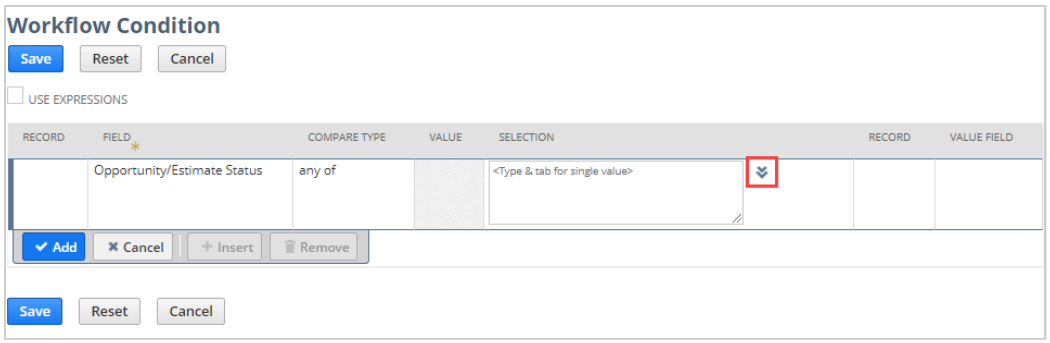
1. If you have not already done so, complete Step 2 Define Workflow Initiation.
2. On the workflow definition page, make sure the Visual Builder is selected and click the open icon to open the Condition Builder. The following screenshot shows the location of the open icon:



   The **Workflow Condition** window appears.
3. In the **Workflow Condition** window, enter the following column information:

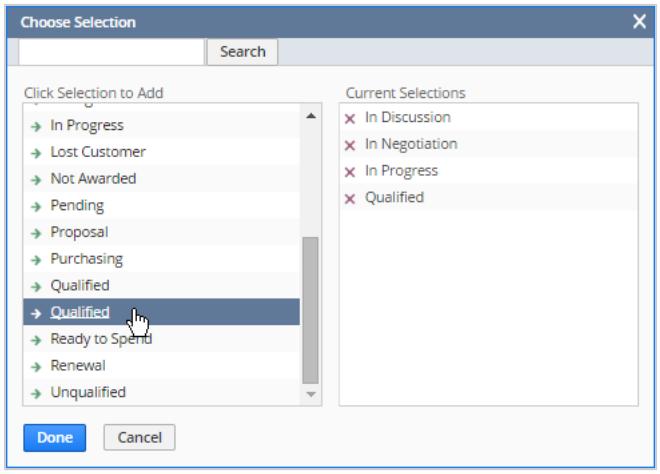
Property	Value
Field	Opportunity/Estimate Status
Compare Type	any of

4. In the **Selection** column, click the Selection box and then click the **Select Multiple** button.

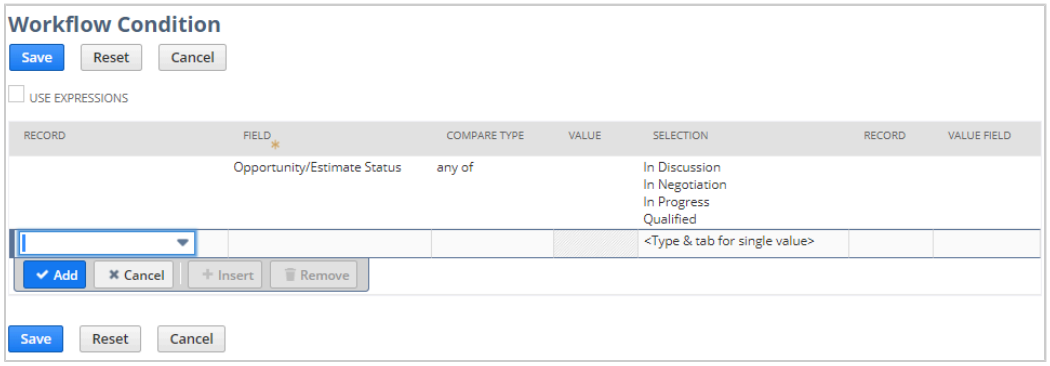


The **Choose Selection** window appears.

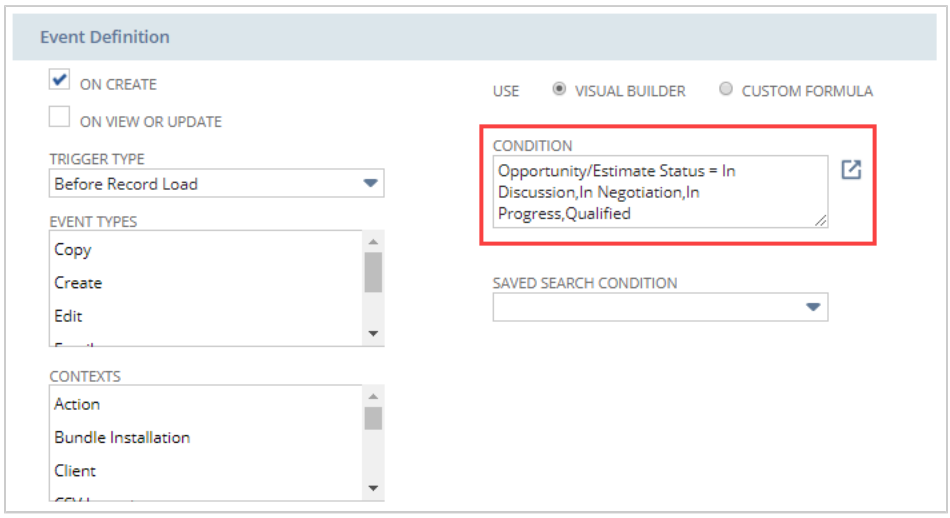
5. In the **Choose Selection** window, in the **Click Selection to Add** column, select **In Discussion**, **In Negotiation**, **In Progress**, and **Qualified**.



6. Click **Done**.
7. In the **Workflow Condition** window, click **Add**.



8. Click **Save**. Verify that the workflow properties look like the following screenshot:



9. Click **Save** to save the workflow. The workflow diagrammer and context panel appears.
10. To continue with the tutorial, continue to Step 4 Create Workflow States.

Step 4 Create Workflow States

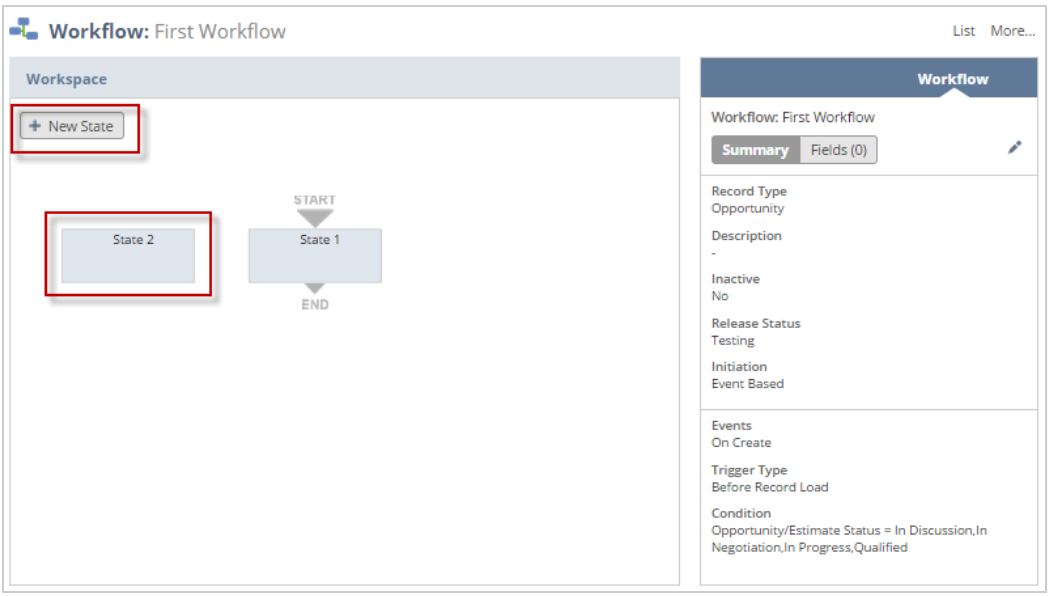
This step shows how to use the workflow diagrammer to create and edit states in a workflow. You must create states before you can create actions or transitions.

When you create a workflow, the Workflow Manager includes a single state by default. In this step you modify the default state and create an additional state. The workflow enters these two states when it runs on an opportunity.

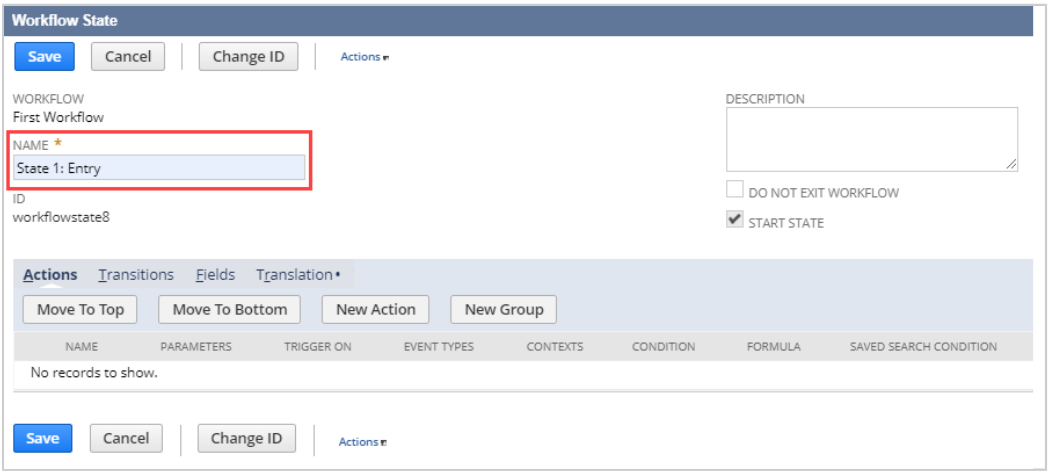
To create the required states for the workflow:

1. If you have not already done so, complete Step 3 Define the Workflow Condition.
2. Create a second state by clicking the **New State** button in the diagrammer.

   The second state, State 2, appears in the diagrammer. The workflow diagrammer already contains the default state, State 1. The **Start** and **End** icons indicate that State 1 is both an entry (Start) and an exit (End) state.



3. To change the name of the State 1, double-click the state in the diagrammer. In the **Name** box of the **Workflow State** window that appears, change the name to **State 1 Entry** and click **Save**.



4. Repeat the previous step to change **State 2** to **State 2 See Opportunities**.

> **Note:** You can also select a state in the diagrammer and click the **Edit** icon in the context panel.

5. To continue with the tutorial, continue to Step 5 Create Actions

Step 5 Create Actions

This step shows you how to use the **Workflow State** window to add actions to a state. The actions added to the states execute according to their triggers when the record enters the state in the workflow.

In this step, you add a Set Field Mandatory action in **State 1 Entry** and a Go To Page action in **Step 2 See Opportunities**.

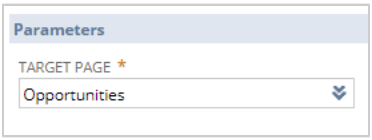
To create the actions in the workflow states:

1. If you have not already done so, complete Step 4 Create Workflow States.
2. To create a new action for **State 1 Entry**, in the workflow diagrammer, double-click **State 1 Entry** and click the New Action button in the **Workflow State** window. The **New Action** window appears.
3. Click **Set Field Mandatory**.
4. In the **Workflow Action** window, set the following properties:

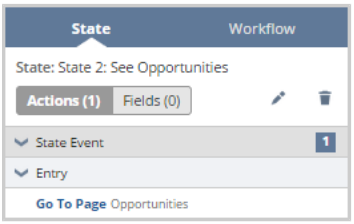
Property	Value
Trigger On	Entry
Field, in the **Parameters** section	Title
Mandatory, in the **Parameters** section	Checked

5. Click **Save**.
6. Verify that the action in the context panel looks like the following screenshot:

7. Repeat step 2 to access the **New Action** window for **State 2 See Opportunities**.
8. Click **Go To Page**. The **Go To Page** window appears.
9. Set the **Trigger On** field to **Entry**.
10. Under **Parameters**, in the **Target Page** field, enter **Opportunities** and select **Opportunities** in the dropdown list that appears. The following screenshot shows **Opportunities** selected in the **Target Page** field:



11. Click **Save**.
12. Verify that the action in the context panel looks like the following screenshot:



13. To continue with the tutorial, continue to Step 6 Create a Transition.

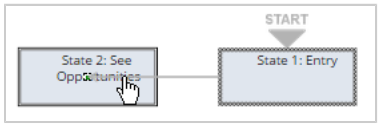
Step 6 Create a Transition

This step shows you how to create and edit a transition. You can use the diagrammer to drag and drop a transition between states and then edit the transition properties. Any changes made in the diagrammer are immediately saved.

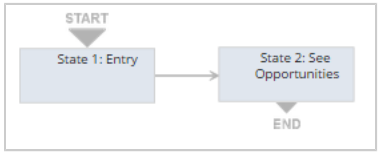
In this step, you create a transition between **State 1 Entry** and **State 2 See Opportunities** that executes after an opportunity record is saved for the first time.

To create and edit a transition:

1. If you have not already done so, complete Step 5 Create Actions.
2. In the diagrammer, hover over the **End** icon on **State 1 Entry**. The cursor becomes a filled half-circle.
3. Drag and hover over **State 2 See Opportunities**. The following screenshot shows how the cursor becomes an arrow and you can drag it to the required state:

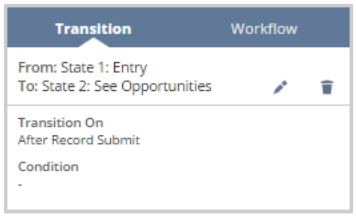


The diagrammer creates a transition when you release the mouse and **State 2 Opportunities** becomes the new exit (End) state. You can also click and drag the states to new locations. The following screenshot shows a transition between state 1 and state 2:



Note: You can also create and edit a transition by editing the state properties for the state that you want to transition from and clicking the **New Transition** button.

4. Select the transition in the diagrammer and click the **Edit** icon in the context panel.
5. In the **Workflow Transition** window, select **After Record Submit** for the **Transition On** property.
6. Click **Save**.
7. Verify that the transition in the context panel looks like the following screenshot:



8. To continue with the tutorial, proceed to Step 7 Initiate and Validate the Workflow.

Step 7 Initiate and Validate the Workflow

This step shows how to initiate the workflow by creating an Opportunity record. After a user creates and saves an opportunity with a status of **In Discussion**, **In Negotiation**, **In Progress**, and **Qualified**, NetSuite initiates an instance of the workflow on the record.

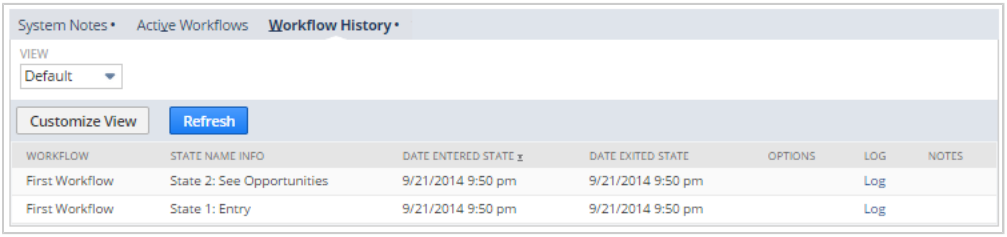
Use the behavior of NetSuite and the workflow activity information to validate the tasks performed by the workflow.

To initiate and validate the First Workflow workflow:

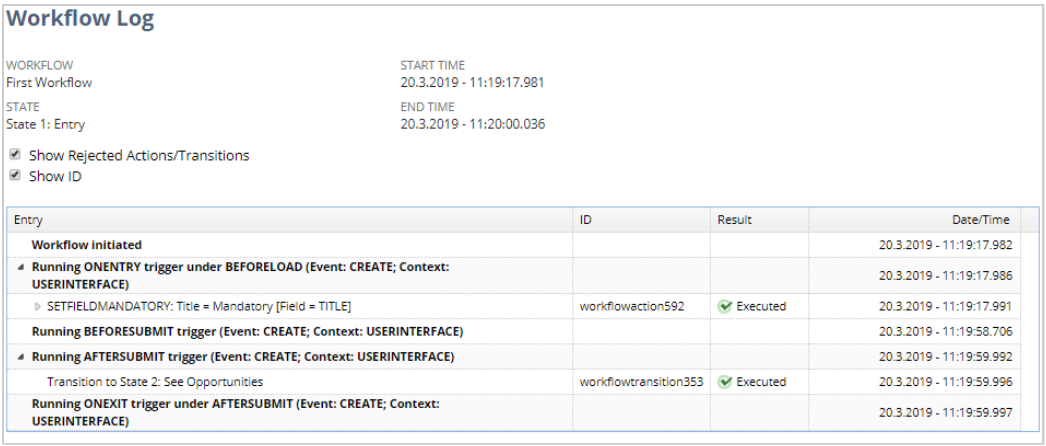
1. If you have not already done so, complete Step 6 Create a Transition.
2. Go to Transactions > Sales > Create Opportunities. A new Opportunity record form opens:
   - The default value of the **Opportunity Status** field on an opportunity is **Qualified**, so the initiation condition for the workflow was met.
   - The **Title** field appears with an asterisk, indicating it is a required field. This indicates that the record entered **State 1 Entry** in the workflow and the Set Field Mandatory action executed.
3. On the Opportunity record form, select a company, enter a value in the **Title** field, and click **Save**.

   The Opportunities list page appears, indicating the After Record Submit trigger executed, the record transitioned to **State 2 See Opportunities**, and the Go To Record action executed.
4. On the Opportunities list page, open the recently created opportunity, click the **System Information** subtab, and then click the **Workflow History** subtab.

   The **Workflow History** subtab shows the two states for the workflow, the date and time each state was entered and exited by the workflow, and the workflow execution log for each state:



5. Click **Log** for **State 1 Entry**. The workflow execution log shows the triggers, actions, and transitions executed:



Note: If the workflow does not execute as expected, you can use the workflow execution log to troubleshoot issues.

Working with Workflows

To create a workflow, you must first define the workflow on the workflow definition page and then use the diagrammer and context panel to create and edit the states, actions, transitions, conditions, and custom fields. For more information about each of the basic elements of a workflow, see Workflow Elements.

To view, create, and edit workflows, you must have the appropriate permission and permission level required for working with the base record types in the workflow. For access to all SuiteFlow functionality, use the Administrator role. For more information, see Required Permissions for SuiteFlow.

The following table describes the basic steps for creating a workflow:

Task	Description
Creating and editing a workflow	Create the workflow on the workflow definition page. Specify the workflow basic properties, event definition, and workflow initiation properties. You can also schedule a workflow, view and edit existing workflows, and copy and inactivate workflows. Optionally, you can create a workflow based on a template. For information, see: ■ Creating a Workflow ■ Creating a Workflow from a Workflow Template ■ Scheduling a Workflow ■ Viewing Existing Workflows ■ Editing a Workflow
Create and editing states	Use the workflow diagrammer to create states, and use the workflow diagrammer, context panel, or **Workflow State** window to edit or delete states in a workflow. See Working with States.
Create and edit actions	Use the context panel or the **Workflow State** window to create, edit, and delete actions. You can also reorder, move, and copy actions, use action groups to use a common condition for actions, or schedule an action. See Working with Actions.
Create and edit transitions	Use the workflow diagrammer or the **Workflow State** window to create, edit, and delete transitions. You can also schedule transitions. See Working with Transitions.
Create conditions	You can create conditions for workflow initiation, or action or transition execution. The conditions can be built with the Condition Builder or you can create formulas with the Formula Builder. See Working with Conditions.
Create custom fields	Create and edit workflow and state fields in the context panel or **Workflow State** window. See Working with Custom Fields.

Creating a Workflow

When you create a workflow, you specify the basic workflow properties, event initiation, and workflow initiation properties. Use the following procedure to create a workflow based on an event. If you want to create a workflow to run on a schedule, see Scheduling a Workflow.

Important: This procedure assumes you are familiar with the elements of a workflow. For more information about each of the workflow elements, see Workflow Elements.

To create a workflow:

1. In NetSuite, go to Customization > Workflow > Workflows > New.
2. Under **Basic Properties**, set the following basic properties:

Property	Description
Name	Name for the workflow. Can be up to 40 characters. Any workflows on the Workflow list page or workflow instances on the **Active Workflows** and **Workflow History** subtabs for the record appear with this name.
ID	Optional script ID for the workflow definition. This value must be lowercase. It cannot include spaces or exceed 26 characters. Use Script IDs if you plan to use the SuiteBundler feature to bundle the workflow and deploy it into another NetSuite account. Script IDs reduce the risk of naming conflicts for workflows deployed into other accounts. NetSuite prepends `customworkflow` to this value when you save the workflow definition. You can change the script ID after you create the workflow. See Editing a Workflow Script ID.
Record Type	Record type on which you want instances of the workflow to run. This record type can be a standard NetSuite record type or a custom record type. You can only specify one base record type per workflow definition. If you select a Customer, Transaction, or Item base record type, you can also select a sub type. See Workflow Base Record.
Sub Type	Secondary record type for Customer, Transaction, or Item record types. Hold down the CTRL key to select multiple sub types.
Description	Optional text description.
Owner	User account that is the owner of the workflow and receives emails for errors that occur during scheduled workflow or scheduled workflow action execution. With a **Release Status** of **Testing**, only the workflow owner can initiate the workflow. See Release Status.
Execute As Admin	Indicates the workflow runs as a user with an administrator role and ignores the role of the user that initiates the workflow. Using this property may give unintended levels of access for the user that initiates the workflow. For more information about this property and usage guidelines, see Execute As Admin.
Release Status	Release status of the workflow. You can select one of the following options: ▪ **Testing**. The workflow only initiates for the **Owner** of the workflow. This is the default Release Status setting for new workflows. ▪ **Released**. The workflow initiates for any user, according to the workflow initiation properties. ▪ **Not Initiating**. The workflow does not initiate, but is visible on the Workflow list page. Any existing workflow instances continue running. ▪ **Suspended**. No new instances of the workflow are created and no existing instances of the workflow are executed. If the workflow includes scheduled workflows, transitions, or actions, none of them will be executed. See Release Status.

Property	Description
Keep Instance and History	Indicates whether the history records for workflows are saved in the database as the workflows run and after they complete. Workflow history records appear in the form of execution and error log records. You can select one of the following options: ■ **Only When Testing**. When you select this option, workflow history records are kept only when the Release Status of the workflow is set to **Testing**. ■ **Never**. When you select this option, workflow history records are never kept. ■ **Always**. When you select this option, workflow history records are always kept. By default, the **Only When Testing** is selected when you create new workflows. When you select the **Never** option, workflow instance and history records are viewable on the record's **Active Workflows** and **Workflow History** subtabs while the workflow is running. When the workflow is finished running or is canceled, the history records related to the workflow instance's execution are deleted from the database. History records are never deleted for workflows that transition to a state with the **Do Not Exit Workflow** box enabled.
Enable Logging	Indicates that NetSuite generates a workflow execution log for each state entered by the workflow during workflow execution. For workflows in testing mode, this property is enabled by default. For workflows in released mode, you must manually enable this property. For more information about the workflow execution log, see Workflow Execution Log.
Inactive	Indicates that no new workflow instances initiate for the workflow and the workflow does not appear by default on the Workflow list page.

3. Under **Initiation**, select **Event Based**.

> **Note:** To create a scheduled workflow, see Scheduling a Workflow.

4. Under **Event Definition**, set the following properties to define the workflow initiation:

Property	Description
On Create | On Update	Specifies that NetSuite initiates a new workflow instance when a record of the base record type is created or updated: ■ **Created.** A record of the base record type is created in NetSuite. ■ **Updated.** A record of the base record type is opened for view or edit in NetSuite. This event can occur in the NetSuite UI or by one of the types of NetSuite functionality listed in the **Context Type** box. For more information, see Workflow Initiation and Execution Contexts and Workflows.
Trigger Type	The server trigger on which you want a workflow instance to initiate. The workflow instance initiates and the record enters the entry state on this trigger type. For more information, see Workflow Triggers and Workflow Initiation Triggers.
Event Types	The activity that resulted in the record being created, viewed, or updated. The workflow instance initiates only if the event occurred. For example, you can limit the workflow to initiate only if a record was copied. For more information, see Workflow Event Types and Event Types Reference.
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.

Property	Description
Condition	Condition to limit the initiation of a workflow instance. A workflow instance for the base record type only initiates if this condition is met. Use the Condition Builder or the Formula Builder to create a condition or formula. For more information, see Defining a Condition with the Condition Builder, Defining a Condition with Expressions, or Defining a Condition with Formulas.
Saved Search	Saved search that returns records of the same type as the base record type for the workflow. NetSuite runs the saved search and initiates a workflow instance if the record meets the criteria in the saved search. The saved search must have at least one filter criteria set on the **Criteria** subtab of the saved search. **Note:** An error can occur if the Saved Search filter is not public and not shared with the user or role. Ensure that saved searches are available to the users and roles that execute the workflow

5. Optionally, if custom fields of type **Workflow** exist, set the field values under **Custom**.

   The **Custom** section does not appear of there are no custom fields of type **Workflow** in the NetSuite account.

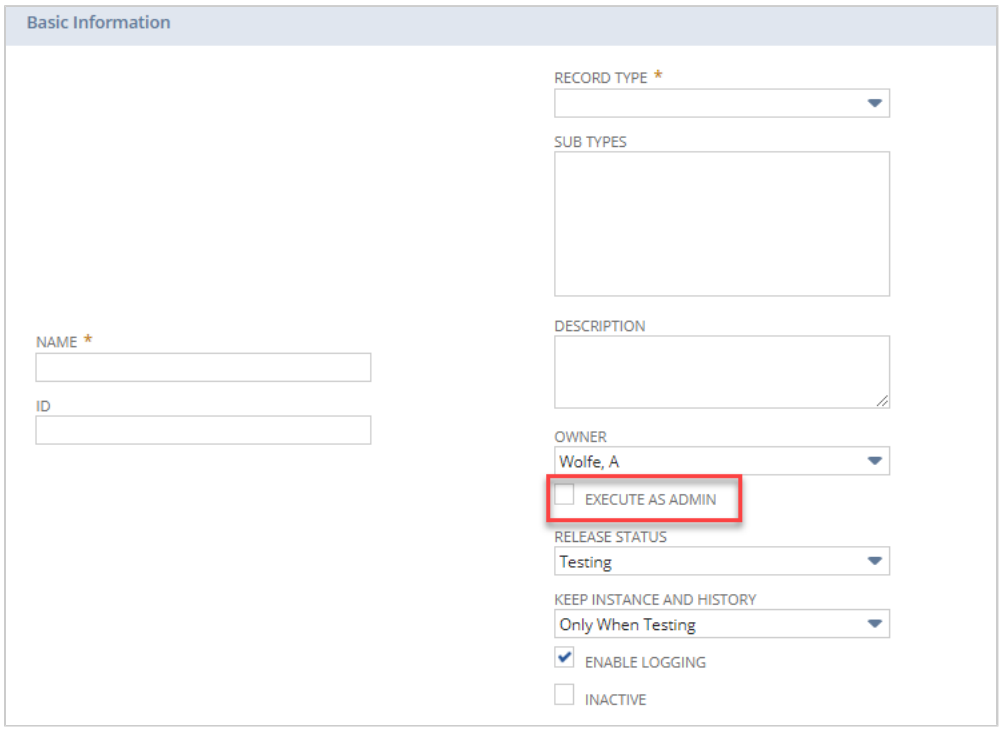
6. Click **Save**. NetSuite creates the workflow with an initial state and opens the workflow in the Workflow Manager.

   You can continue to define the workflow by adding states. See Working with States.

Note: If you do not want the workflow to run, set it to inactive or set the release status to **Not Initiating** or **Suspended** before you save it. See Inactivating a Workflow and Release Status.

Execute As Admin

On the workflow definition page, you can select the **Execute As Admin** property to direct NetSuite to execute workflow instances as a user with the administrator role. This administrator role privileges override the privileges for the user role that initiated the workflow instance.



Some types of workflows may require administrator privileges. For example, you lay out a workflow that creates follow-up tasks after a sales order is saved and the workflow instance needs to read data from employee records. The workflow instance does not execute properly if the user that initiated the workflow instance does not have permission to access employee records.

Rules and Guidelines for Using Execute As Admin

Important: Use **Execute As Admin** only when required. Otherwise, you may give unintended data access to users for whom this workflow initiates.

Use the following rules and guidelines with this property:

- If you enable **Execute As Admin**, any action that includes a record join must execute on a server trigger and not a client trigger. Any action that executes on a client trigger will not execute as expected. For more information about the difference between server and client triggers, see Workflow Triggers.
- Only users with the administrator role have access to **Execute As Admin**. This property is disabled for all other roles, including roles with the Workflow permission assigned with the full permission level. This is also true for **Execute As Admin** on the SuiteScript Deployment page.
- If you enable **Execute As Admin**, the Override Period Restrictions permission is not automatically enabled for the workflow. Instead, the logged in user's value for this permission is applied to the workflow, to determine whether the workflow can complete updates to Posting Period field values. This exception is intended to prevent users who do not have the Override Period Restrictions permission from executing workflows that post transactions in locked periods. In addition, other updates made by a workflow with **Execute As Admin** enabled, such as changes to the **Approval Status** field, may still lead to changes in Posting Period field values, even if the logged in user does not have the Override Period Restrictions permission.
- The Execute As Admin feature does not impact the Allow Non G/L Changes permission associated with editing period end journals. When you select the Allow Non G/L Changes box, users with the

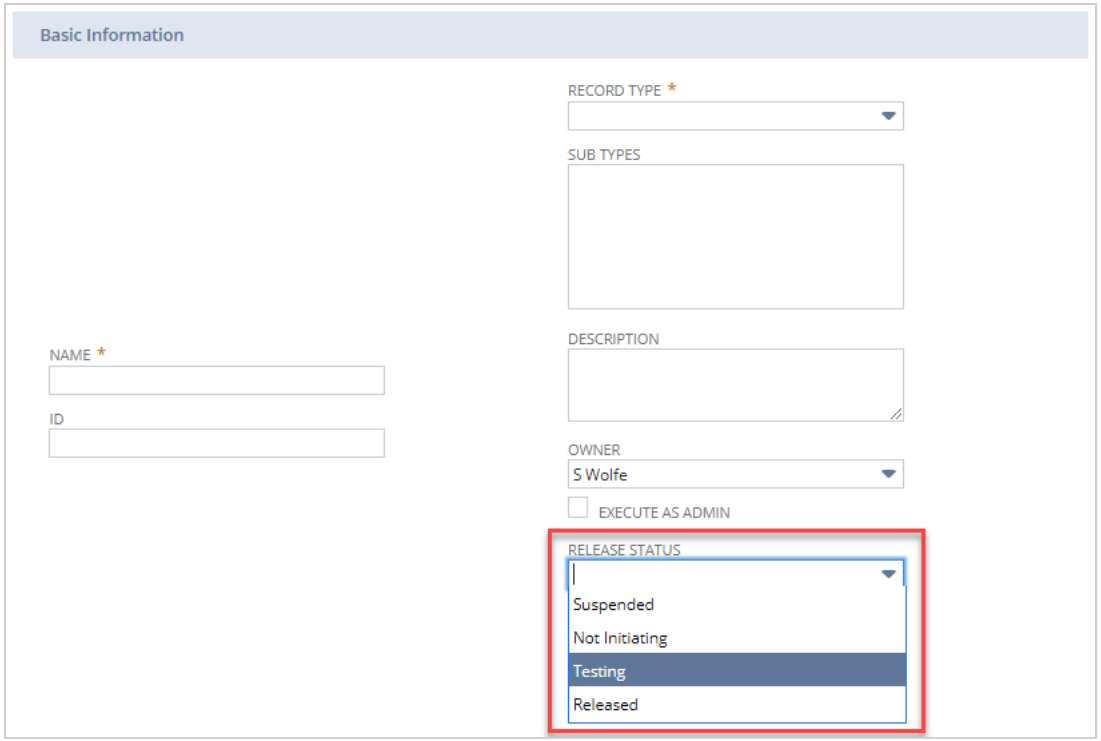
Allow Non G/L changes permission assigned to their role can make edits that do not affect the general ledger. If you enable **Execute As Admin**, the editing permissions of period end journals does not change.

- If you create a workflow that is initiated by users accessing records through the Employee Center, you will commonly need to enable **Execute As Admin**.

Note: If you create a workflow that includes an Add Button action, when a user interacts with the button and the record is saved, the System Notes will list the role of the user as Administrator, regardless of the role the user is logged in as. This behavior is expected and is not related to the Execute As Admin property.

Release Status

Use the Release Status property of the workflow to test the workflow, to make it available for initiation by all users, or to keep it from running.



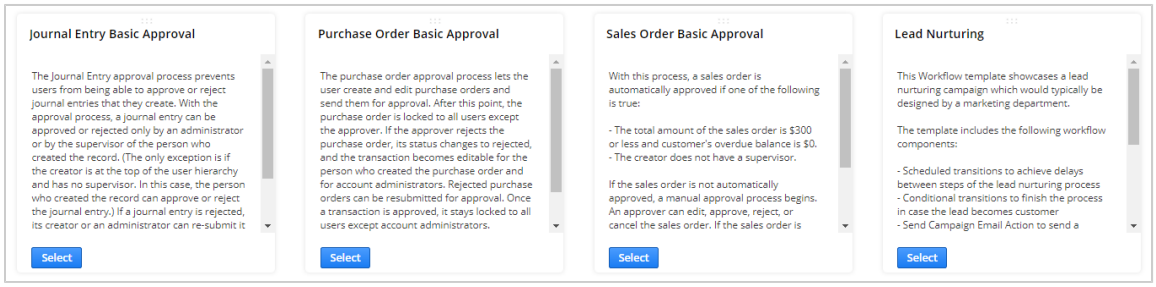
The following table describes the workflow release statuses and when you should use each status:

Release Status	Description	When to Use
Suspended	No new instances of the workflow are created and no existing instances of the workflow are executed. If the workflow includes scheduled workflows, transitions, or actions, none of them will be executed. It is not possible to initiate a suspended workflow from SuiteScript, or by using the Initiate Workflow or Mass Update actions in SuiteFlow.	Use when you want a workflow to stop executing but you do not want to delete the workflow.

Release Status	Description	When to Use
	To resume a suspended workflow, change the Release Status of the workflow to a different release status than Suspended.	
Not Initiating (Not initiating mode)	New instances of the workflow do not initiate but already running instances of the workflow will continue to run. The workflow still appears on the Workflow list page. Similar to inactivating a workflow.	Use when you do not want any new instances of the workflow initiated but you still want existing instances of the workflow to execute. See Viewing Existing Workflows.
Testing (Testing mode)	A workflow instance only initiates for the owner of the workflow. However, any user accessing the record will see changes made to the record by the workflow. Workflow actions execute for the user even though the workflow instance was not initiated by the user. Scheduled workflows do not execute when the workflow is in the testing mode. This is the default Release Status option when you create new workflows.	Use when you want to test a workflow. In testing mode, the workflow execution log appears for each state on the **Workflow History** subtab for any record in a workflow. See Workflow Execution Log.
Released (Released mode)	An instance of the workflow initiates for any user with access to the base record type of the workflow definition. Scheduled workflows, actions, and transitions only execute when the workflow is in released mode. Enable logging on the workflow definition if you want to continue generating workflow execution logs.	Use when you want the workflow to run for all users. For more information about which users can initiate a workflow, see Users Who Can Run Workflows and Defining the Workflow Audience.

Creating a Workflow from a Workflow Template

Use workflow templates as a starting point to create customized workflows to define typical or common business processes. To create a workflow from a template, select the template and then edit the workflow definition properties, states, actions, and transitions to customize the workflow according to your business needs.



Important: This procedure assumes you are familiar with the elements of a workflow. For more information about each of the workflow elements, see Workflow Elements. For more information about workflow templates, see Workflow Templates Reference.

To create a workflow from a template:

1. In NetSuite, go to Customization > Workflow > Workflows > New.
2. Click **From Template**.
3. Click **Select** for one of the following workflow template types:
   - Journal Entry Basic Approval Template
   - Purchase Order Basic Approval Template
   - Sales Order Basic Approval Template
   - Lead Nurturing Template

NetSuite creates the workflow in the Workflow Manager. You can then edit the workflow definition properties, states, actions, and transitions.

For more information about editing a workflow after you create it from a template, see Editing a Workflow.

Note: When you create a workflow from a template, the default value for the Release Status is **Not Initiating**. You must change the release status to **Testing** to test the workflow. See Release Status.

Scheduling a Workflow

When you configure a workflow to run on a schedule, the workflow executes on the results of a saved search. When you choose to initiate a workflow on a scheduled basis, you select a saved search and then define the schedule on which the workflow runs. You can schedule the workflow to run one time, every 30 minutes, or at a certain time on a daily, weekly, monthly, or yearly basis.

Before you define the schedule for the workflow, you must create the saved search. The saved search must run on the same record type as the base record type for the workflow. The saved search must also include at least one filter set on the **Criteria** subtab for the saved search. On the workflow definition page, select the **Scheduled** option, select the saved search, and set the schedule for the workflow.

At the scheduled time, NetSuite runs the saved search and initiates a workflow instance to run on each of the records in the search results. For example, if the saved search returns 10 records, NetSuite initiates 10 instances of the workflow. Scheduled workflows always run on all of the records in the search results, not only the records you can see on the main page (for summary type result) of the saved search.

To create a workflow that runs on a schedule, you select the saved search on which to run the workflow and select the schedule options. You can schedule the saved search and workflow to run a single time or you can select the **Repeat** option to run the saved search and workflow on a repeating schedule. If you want the workflow to run one time, select the saved search, disable the **Repeat** option, and select the **Execution Date** and **Execution Time**. If you want the workflow to run more than one time, select the saved search, enable the **Repeat** option, and set the related options.

Consider the following when you create scheduled workflows:

- NetSuite runs multiple workflows in parallel.
- There is no inherent order in which scheduled workflows start and finish.
- Only scheduled workflows with a release status of **Released** execute on a schedule.
- All workflow initiation criteria must be specified when you create the saved search. You cannot set a condition to initiate a scheduled workflow.

For more information about how NetSuite runs scheduled workflows, see Initiating a Workflow on a Schedule and Scheduled Trigger.

To create a scheduled workflow:

1. Define a saved search on the base record type for which you want to create a workflow.

   The saved search must also include at least one filter set on the **Criteria** subtab for the saved search. For more information, see the help topic Defining a Saved Search.
2. In NetSuite, go to Customization > Workflow > Workflows > New. The workflow definition page appears.
3. Under **Basic Properties**, set the following basic properties:

Property	Description
Name	Name for the workflow. Can be up to 40 characters. Any workflows on the Workflow list page or workflow instances on the **Active Workflows** and **Workflow History** subtabs for the record appear with this name.
ID	Optional script ID for the workflow definition. This value must be lowercase. It cannot include spaces or exceed 26 characters. Use script IDs if you plan to use the SuiteBundler feature to bundle the workflow and deploy it into another NetSuite account. Script IDs reduce the risk of naming conflicts for workflows deployed into other accounts. NetSuite prepends `customworkflow` to this value when you save the workflow definition. You can change the script ID after you create the workflow. See Editing a Workflow Script ID.
Record Type	Record type on which you want instances of the workflow to run. This record type can be a standard NetSuite record type or a custom record type. You can only specify one base record type per workflow definition. If you select a Customer, Transaction, or Item base record type, you can also select a sub type. See Workflow Base Record.
Sub Type	Secondary record type for Customer, Transaction, or Item record types. Hold down the CTRL key to select multiple sub types.
Description	Optional text description.
Owner	User account that is the owner of the workflow and receives emails for errors that occur during scheduled workflow or scheduled workflow action execution. With a **Release Status** of **Testing**, only the workflow owner can initiate the workflow. See Release Status.
Execute As Admin	Indicates the workflow runs as a user with an administrator role and ignores the role of the user that initiates the workflow. Using this property may give unintended levels of access for the user that initiates the workflow. For more information about this property and usage guidelines, see Execute As Admin.
Release Status	Release status of the workflow. You must set the status to **Released** for a workflow to run on a schedule. However, with a release status of **Testing**, you can execute a scheduled workflow immediately using the **Execute Now** button. See Testing Scheduled Workflows.
Keep Instance and History	Indicates whether the history records for workflows are saved in the database as the workflows run and after they complete. Workflow history records appear in the form of execution and error log records. You can select one of the following options: ■ **Only When Testing**. When you select this option, workflow history records are kept only when the Release Status of the workflow is set to **Testing**. ■ **Never**. When you select this option, workflow history records are never kept. ■ **Always**. When you select this option, workflow history records are always kept. By default, the **Only When Testing** is selected when you create new workflows. When you select the **Never** option, workflow instance and history records are viewable on the record's **Active Workflows** and **Workflow History** subtabs while the workflow is

Property	Description
	running. When the workflow is finished running or is canceled, the history records related to the workflow instance's execution are deleted from the database. History records are never deleted for workflows that transition to a state with the **Do Not Exit Workflow** box enabled. **Note:** This setting only impacts newly created history records. If you change the Keep Instance and History setting, existing workflows will not be impacted.
Enable Logging	Indicates that NetSuite generates a workflow execution log for each state entered by the workflow during workflow execution. For workflows in testing mode, this property is enabled by default. For workflows in released mode, you must manually enable this property. For more information about the workflow execution log, see Workflow Execution Log.
Inactive	Indicates that no new workflow instances initiate for the workflow and the workflow does not appear by default on the Workflow list page.

4. Under **Initiation**, select **Scheduled** and select the saved search you created in Step 1 in the **Saved Search** dropdown list.

Note: To create an event-based workflow, see Creating a Workflow.

5. If you want the schedule to repeat, enable **Repeat**.
6. Enter the following schedule options:

Option	Description
Frequency	The frequency with which you want the scheduled workflow to run. You can select one of the following options: - Every 30 minutes - Daily - Weekly - Monthly - Yearly If you select a daily, weekly, monthly, or yearly schedule, you need to set additional options to define the schedule. Default is every 30 minutes.
Scheduled From Date	Date on which the schedule starts. The date of the first run depends on the schedule type and schedule options.
Scheduled Until Date	Date on which the schedule ends. The date of the last run depends on the schedule type and schedule options.
Execution Time	Time of day when NetSuite runs the saved search and executes the workflow on the results. Default is midnight.

7. If you select a daily, weekly, monthly, or yearly schedule, set one of the following related options:

Schedule Type	Options
Daily	Set the interval for the number of days between each scheduled execution, or set execution to repeat on every week day.
Weekly	Set the interval for the number of weeks between each scheduled execution and set the day(s) of the week.

Schedule Type	Options
Monthly	Set the specific day in the month (for example, 8th day) or the day of the week in the month (for example, second Tuesday). For both options, you can also set the time period (for example, every 2 months). You can set the following options for the occurrence of the day: ▪ first ▪ second ▪ third ▪ fourth ▪ last
Yearly	Set the specific day in a specific month (for example, 8th of September) or the day of the week in a specific month (for example, second Tuesday in September). You can set the following options for the occurrence of the day: ▪ first ▪ second ▪ third ▪ fourth ▪ last

8. Click **Save**. NetSuite creates the workflow and opens the workflow in the Workflow Manager.

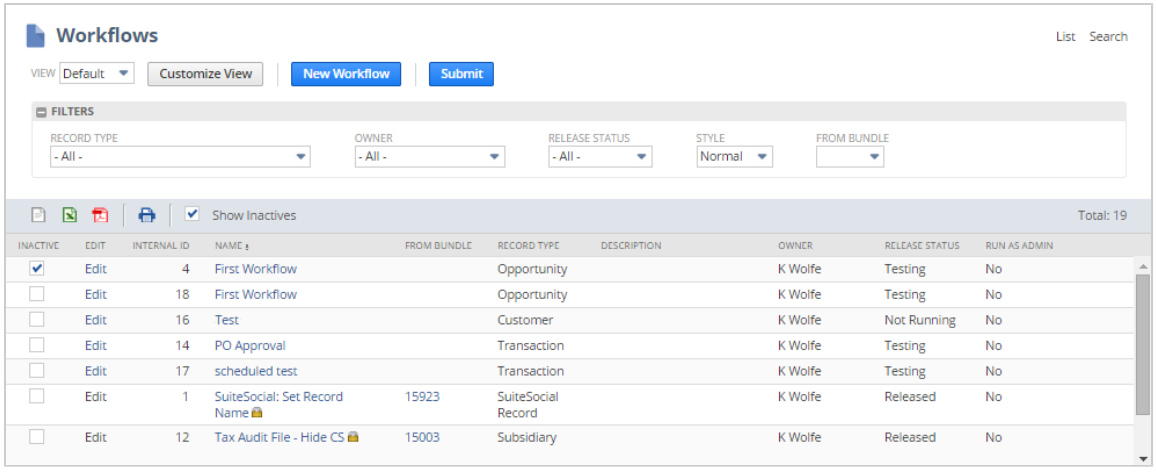
   You can continue to define the workflow by adding states. See Working with States.

Viewing Existing Workflows

Use the Workflows list page to view existing workflows. By default, the list page shows all workflows except workflows marked as inactive. To view inactive workflows, check the **Show Inactives** box.

To access a workflow on the Workflows page, you must have the full permission level assigned for the base record on which the workflow was built. Further, you must have the feature related to the base record enabled in your account.

The following screenshot shows the Workflows list page:

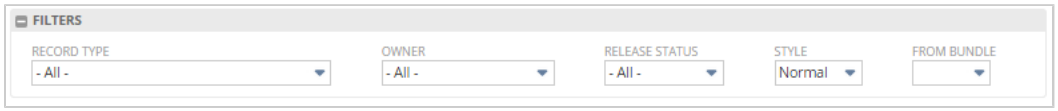


The following table describes the tasks you can complete from the Workflow list page:

Task	Description
Create a new workflow	Click **Create Workflow** to open the workflow definition page. See Creating a Workflow and Scheduling a Workflow.
Edit a workflow	Click **Edit** to edit a workflow. See Editing a Workflow.
Search for workflows	Click **Search** to search for workflows definitions. See Workflow Searches.
Filter the workflows list	Use the **Filters** section to show workflows with specific properties. See Filtering the Workflow List Page.
Mark workflows as inactive	Use the **Inactive** column to mark workflows as inactive. See Inactivating a Workflow.
Customize the workflow list page	Customize the columns that display on the Workflow list page. See the help topic Customizing Sublist Views.

Filtering the Workflow List Page

Use the **Filters** list on the Workflows list page to refine the workflows shown to find a specific workflow or types of workflows.



The following tables describes the available filters:

Filter Name	Description
Record Type	Base record type for a workflow. If you want to view workflows for the Customer, Transaction, or Item record types, the list shows all workflows based on sub types of these records. You cannot filter by sub type. See Workflow Base Record.
Owner	User account set for the **Owner** property on the workflow.
Release Status	Release status of the workflow. You can select **Not Initiating**, **Suspended**, **Testing**, and **Released**. See Release Status.
Style	Style for the display. Select **Normal**, **Grid**, or **Report**. Default view for the Workflows list page is **Normal**.
From Bundle	If your account contains workflows installed as part of a bundle, this list shows all bundle IDs. Use this filter to find all workflows installed from a specific bundle. You can select **Any**, **None**, or select a single bundle ID.

To filter the workflows on the Workflow list page:

1. If you have not already done so, go to Customization > Workflow > Workflows.
2. Expand the **Filters** section.
3. Select the appropriate filters.

   The Workflows list page automatically updates with the appropriate filters.

Editing a Workflow

You can edit an existing workflow to change the workflow properties or elements, or create new ones. When you edit a workflow, you can make changes to the following workflow elements:

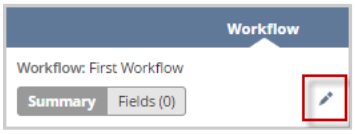
- Workflow definition –includes basic properties, initiation properties, and conditions. The changes you make to workflow initiation conditions or workflow definition properties only affect future instances of the workflow. The changes do not affect currently running workflow instances.
- States –includes creating, editing, and deleting actions, transitions, and custom fields.
- Actions –includes action properties and conditions.
- Transitions –includes transition properties and conditions.
- Workflow and state fields.

You can also edit a workflow definition to change the script ID for the workflow. See Editing a Workflow Script ID. In addition, you can copy workflows and inactivate workflows. For more information, see Copying a Workflow and Inactivating a Workflow.

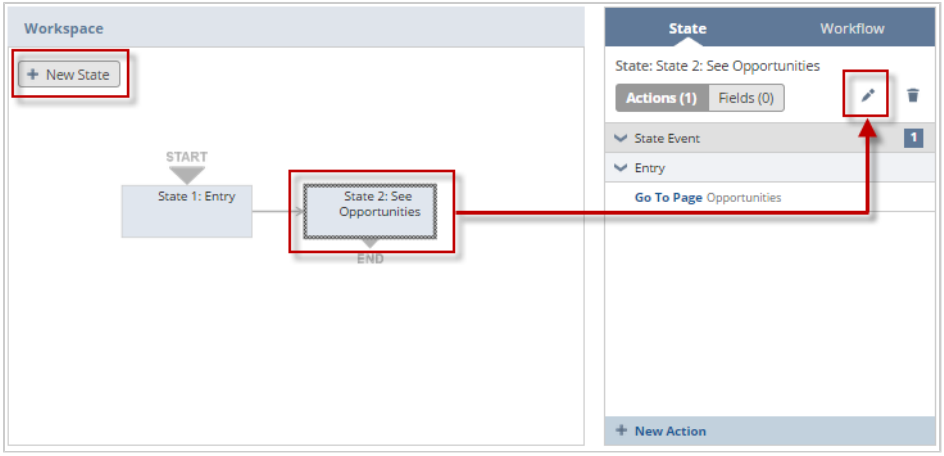
Note: You cannot change the base record type for a workflow after you create it. You can only edit the sub types for the base record type.

To edit a workflow:

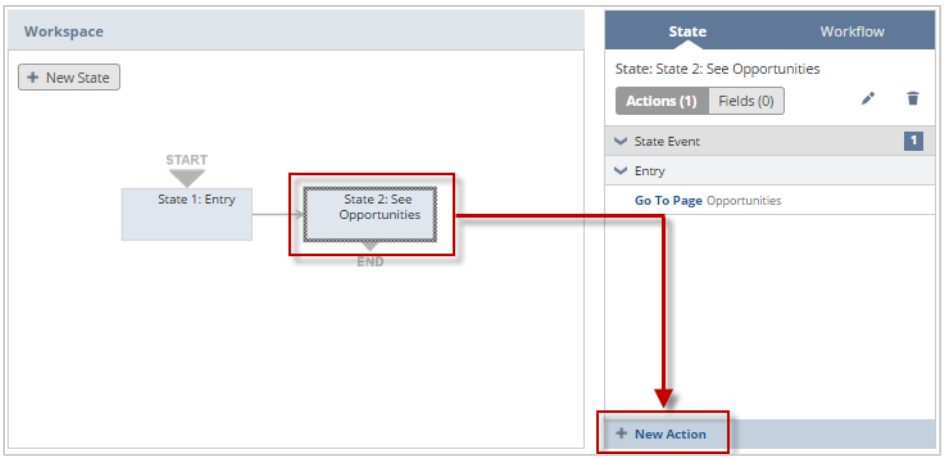
1. Go to Customization > Workflow > Workflows.
2. Locate the workflow that you want to edit and click **Edit**.
3. Do any of the following:
   - To edit the workflow definition, click the **edit** icon on the **Workflow** tab in the context panel. On the workflow definition page, edit the workflow definition and click **Save**.



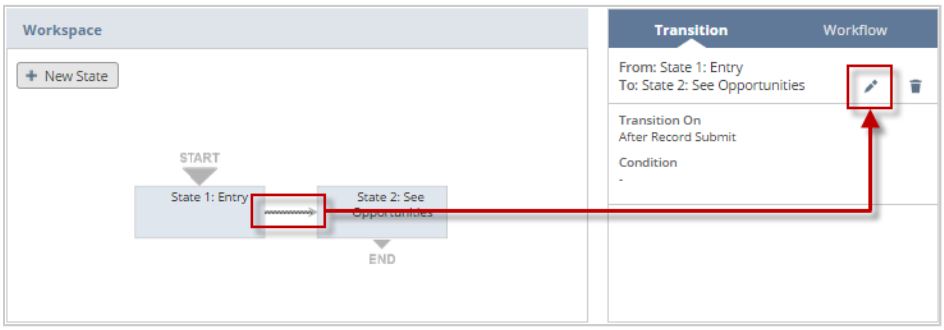
   - To edit a state, select the state and click the **edit** icon on the **State** tab in the context panel. In the Workflow State window, edit the state properties and click **Save**.



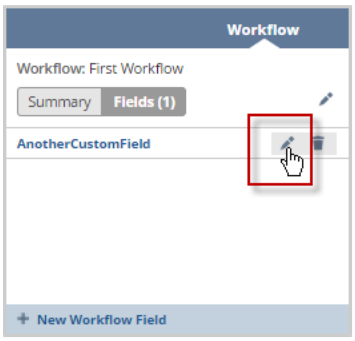
   - To edit an action, select the state that contains the action that you want to change. On the **State** tab in the context panel, hover over the action and click the **edit** icon. On the Workflow Action page, edit the action properties and click **Save**.



- To edit a transition, select the transition that you want to change. On the **Transition** tab of the context panel, click the **edit** icon. On the Workflow Transition page, edit the transition properties and click **Save**.



- To edit a workflow field, click the **Fields** tab on the context panel. Click the **edit** icon. Make your changes and click **Save**.
- To edit a state field, select a state in the diagrammer, and click the **Fields** tab on the context panel. Click the **edit** icon. Make your changes and click **Save**.



Editing a Workflow Script ID

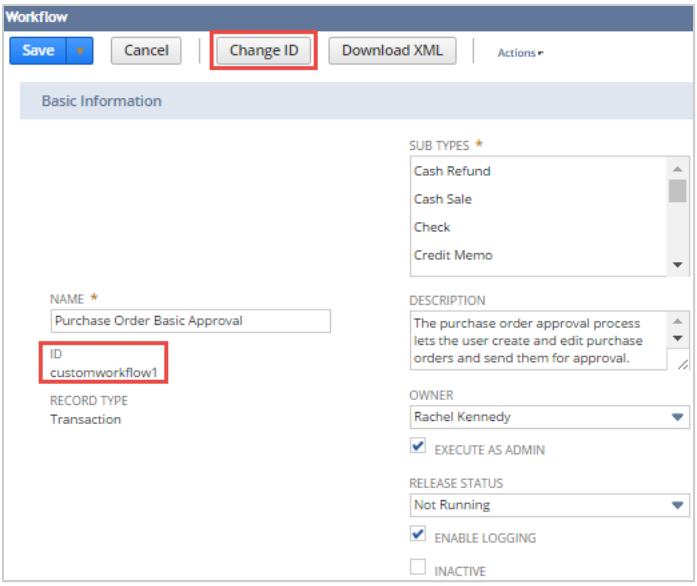
After you create a workflow, you can edit the script ID for the workflow on the workflow definition page. You can edit the ID to change the script ID that you specified when you created the workflow, or override the script ID automatically generated by NetSuite.

To edit the script ID for a workflow definition:

1. If you have not already done so, go to Customization > Workflow > Workflows to open the Workflow list page or search for a workflow.

   See Viewing Existing Workflows and Workflow Searches.

2. Click **Edit** to open the workflow in the Workflow Manager.
3. On the workflow definition page, click **Change ID**.



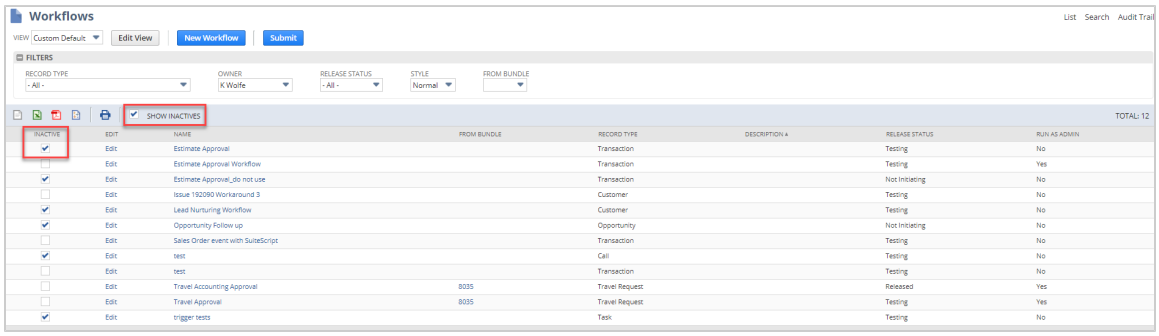
4. In the **Change Script ID** window, enter a new scipt ID. You can prepend an underscore (_) for readability.

5. Click **Save**.

Inactivating a Workflow

You can set a workflow to inactive on the workflow definition page or on the Workflows list page. Inactivate a workflow if you do not want it to execute or appear by default in the Workflows list page. You can view inactive workflows by checking the **Show Inactives** box on the Workflows list page.

The effects on the workflow is the same as a workflow with a release status of **Not Initiating**. For more information about release statuses, see Release Status.



Note: You can also enable the **Inactive** checkbox on the workflow definition page to inactivate a workflow.

To inactivate a workflow:

1. Go to Customization > Workflow > Workflows.
2. On the Workflows list page, enable **Show Inactives**.
3. Optionally, set a filter. See Filtering the Workflow List Page.
4. Enable the **Inactive** checkbox for the workflow that you want to inactivate.
5. Click **Submit**.

Note: This change does not affect currently running instances of the workflow, only new instances.

Copying a Workflow

You can copy a workflow in the Workflow Manager. Create a copy of a workflow to replicate complicated workflows for testing purposes or as a basis to start creation of a new workflow. Use the Make Copy command, available from the **More** menu in the Workflow Manager, to copy a workflow.

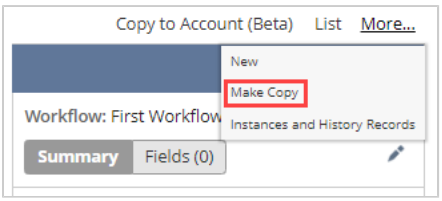
When you copy a workflow, NetSuite makes the following changes to the copy of the workflow:

- Appends a number to the workflow name. For example, the workflow name **First Workflow** becomes **First Workflow (2)**.
- Appends an underscore (_) and number to the workflow script ID and any workflow or state field. For example, a workflow script ID of **customworkflow_firstworkflow** becomes **customworkflow_firstworkflow_2**.
- Sets the release status of the new workflow to **Not Initiating**.

To create a copy of a workflow:

1. If you have not already done so, go to Customization > Workflow > Workflows.

   You can also search for a workflow. See Workflow Searches.
2. Click **Edit** to open the workflow in the Workflow Manager.
3. Click **More** and select **Make Copy**.



NetSuite opens the copy of the workflow in the Workflow Manager.

4. Edit the copy of the workflow. See Editing a Workflow.

Using SDF and Copy to Account

You can use SuiteCloud Development Framework (SDF) to manage custom objects as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview.

You can use the Copy to Account feature to copy a workflow to another of your accounts. To copy a workflow, in the Workflow Manager, click **Copy to Account**.

For information about Copy to Account, see the help topic Copy to Account Overview.

Working with States

A workflow can contain as many states as determined by the business process implemented in the workflow. Every NetSuite workflow must contain at least one state. Workflows containing no states cannot be initiated to run for a record, even if the workflow is set to Running. A workflow can have only one start state, called the entry state. It can have multiple end states, called exit states. A workflow with only one state is also known as a single state workflow.

For an overview of states and state functionality in SuiteFlow, see Workflow States and SuiteFlow Trigger Execution Model.

After you create a workflow, the Workflow Manager creates a default state, State 1. You can edit this state to change the properties and add actions, transitions, and state fields. You can also create additional states or delete existing states. A workflow must have at least two states before you can create transitions. You can create all the states required by the business process first, and then edit the states and create transitions between the states.

Note: As you lay out the workflow and add states, you can use the diagrammer to drag and drop states to reflect the flow of the business process.

The following table describes where you can get more information about working with states:

Task	For more information
Creating a state in a workflow	Creating a State
Editing existing state	Editing a State
Deleting a state and the effects on subsequent workflow instances	Deleting a State

Task	For more information
Creating a state field	Creating and Using State Fields
Working with exit states	Exit States
Creating a non-exiting state	Non-Exiting Workflow States

Creating a State

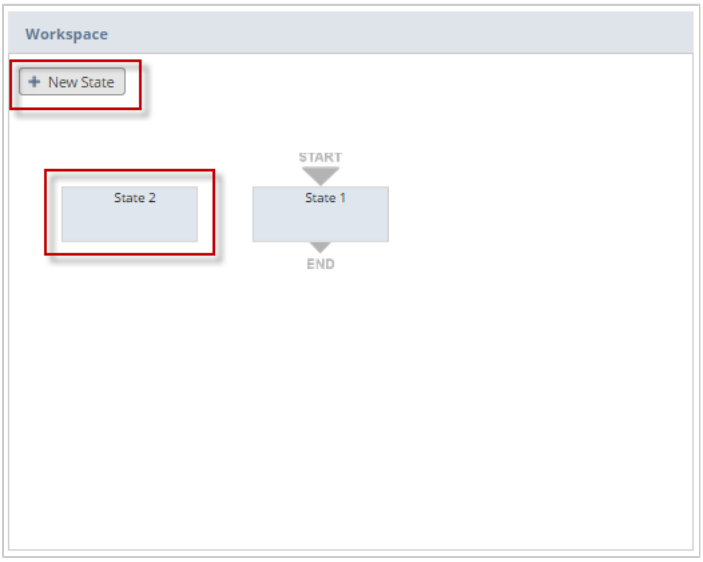
Use the **New State** button in the workflow diagrammer to create a state in a workflow. After you create the state, you edit the state to modify the state properties and create actions and transitions.

Note: When you create a new state, you do not enter any state properties. Edit the state to enter any additional properties. See Editing a State.

To create a workflow state:

1. If you have not already done so, create a new workflow or open an existing workflow by clicking **Edit** on the Workflows page.

   See Creating a Workflow, Viewing Existing Workflows, or Workflow Searches.
2. In the diagrammer or the context panel, click **New State**. The state appears in the diagrammer.



3. Edit the state to change state properties and add actions and transitions. See Editing a State

Editing a State

Edit an existing state to change state properties or create or edit existing workflow elements. Edit a state to make changes to the following workflow elements:

- State properties
- Actions, including action properties and conditions

- Transitions, including transition properties and conditions
- State fields

To edit a state:

1. If you have not already done so, open the workflow that contains the state that you want to edit.
2. Select the state in the diagrammer and click the **Edit** icon on the **State** tab in the context panel.
3. Edit the following state properties:

Property	Description
Workflow	Name of the workflow that contains the state. This field is not editable.
Name	Name of the state. If you edit the name of an existing state, the **Workflow History** subtab shows states that changed names while the workflow runs. See Deleted and Renamed States.
Script ID	Custom script ID for the state. NetSuite prepends `workflowstate` to the value. To edit this field, see Editing a Workflow State Script ID.
Description	Optional description for the state.
Do Not Exit Workflow	Indicates that the workflow instance does not terminate when the workflow completes the actions in this state. If you create a transition from this state to another state, this property is disabled. Also, this property is disabled if there is a transition from this state to another state. For more information about using this property, see Non-Exiting Workflow States.
Start State	If checked, indicates that this is the entry state in a workflow. This field is disabled if it is the current start state for the workflow. If you create a transition to this state, or if you enable this property for another state, NetSuite unchecks this property.

4. To make changes to actions, on the **Actions** subtab, click **Edit** next to an existing action. For more information, see Working with Actions.
5. To make changes to transitions, on the **Transitions** subtab, click **Edit** next to an existing transition. For more information, Working with Transitions.
6. To make changes to custom fields, on the Fields subtab, click the field name to edit it.
7. Click **Save** save the changes to the state.

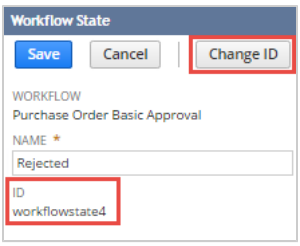
Editing a Workflow State Script ID

When you create a state, NetSuite automatically assigns the state a script ID. Change the script ID to use the script ID as a reference to the state in a script.

The script ID must be must be lowercase. It cannot include spaces or exceed 27 characters. NetSuite prepends `workflowstate` to the value when you save the changes.

To edit the script ID for a workflow state:

1. If you have not already done so, open the workflow that contains the state that you want to edit.
2. Select the state in the diagrammer and click the **Edit** icon on the **State** tab in the context panel.
3. Click **Change ID.**



4. In the **Change Script ID** window, enter a new scipt ID. You can prepend an underscore (_) for readability.

5. Click **Save**.

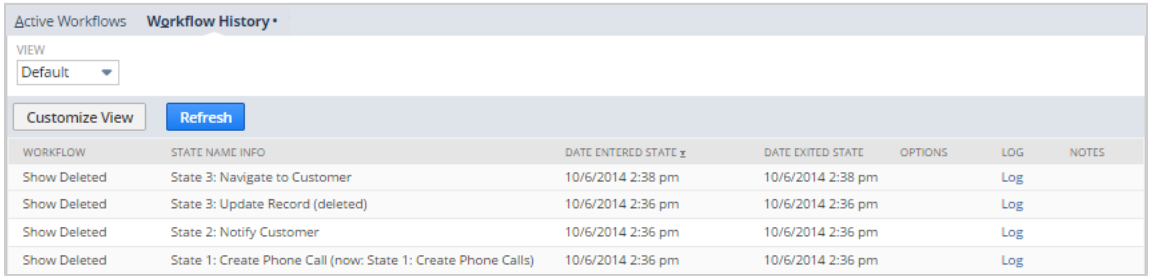
Deleting a State

Delete a state to remove it from the workflow. You can delete a state on a workflow with any release status, if any record in an active workflow instance is not currently in the state. In addition, if you delete a state with transitions assigned to it, the transitions are also deleted.

Note: Setting the release status of a workflow instance to Testing or Not Initiating or changing a workflow definition to Inactive does not cancel actively running workflows. The current state of any running workflow instance cannot be deleted.

For any instance of the workflow, the workflow history displays any state that was deleted after the workflow entered the state. The workflow history displays the state for any currently running or completed workflow. The state appears with **(deleted)** appended to it. To view the workflow history, go to the record that is associated with the workflow and click the **Workflow History** subtab under the **System Information** tab.

The following screenshot shows a workflow history with a deleted state:



To delete a state:

1. If you have not already done so, open the workflow that contains the state that you want to edit.

2. In the diagrammer, select the state that you want to delete.
3. Click the **Delete** icon on the **State** tab of the context panel. Any incoming or outgoing transitions are also deleted.

> **Note:** You can also open the **Workflow State** window, and then from the Actions menu, select Delete.

4. Click **Yes** in the popup that appears.

Working with Actions

Each state in a workflow can contain actions. The actions used in a workflow depend on the type of business processes automated by the workflow. Each state can contain one or more actions. Each action includes a basic set of properties that apply to all actions. In addition, each action has specific properties, based on the values of the record in a workflow, that you use to configure the action. For an overview of actions in SuiteFlow, including action properties, triggers, and conditions, see Workflow Actions.

For a list of all actions available in SuiteFlow, see Workflow Actions Overview.

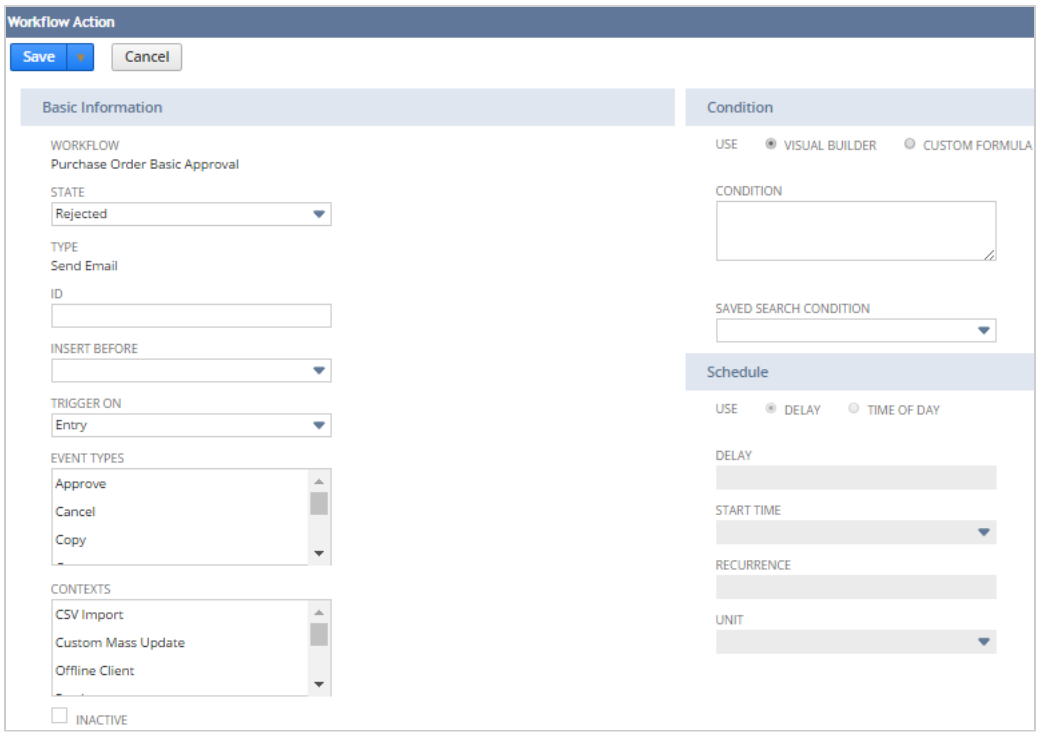
The following table describes where you can get more information about working with actions:

Task	For more information, see ...
Create an action in a state	Creating an Action
Edit an existing action	Editing an Action
Delete an action	Deleting an Action
Order actions in a state	Ordering Actions
Reorder actions in a state	Reordering Actions
Move an action to another state	Moving an Action
Copy an action	Copying an Action
Use an action group to use the same condition with multiple actions	Using Action Groups
Use a sublist action group to execute actions on Items sublist lines	Using Sublist Action Groups
Set an action to execute on a schedule	Scheduling an Action
View a list of all available actions and links to more information	Workflow Actions Overview
Examples of using actions in a workflow, including the Add Button action and executing an action with a saved search as a condition	Action Examples

Creating an Action

Create an action from the **Workflow State** window or the **New Action** button on the context panel. Most of the properties for actions are common to all action types.

The following screenshot shows the common action properties:



Note: When you create an action, the context panel displays the action according to its trigger category. However, actions execute in the order in which they appear on the **Actions** subtab of the **Workflow State** window, according to their trigger type. For more information about how actions execute in a state according to their trigger type, see Action Triggers.

To create an action:

1. If you have not already done so, open the existing workflow that contains the state where you want to create the action.

   See Viewing Existing Workflows and Workflow Searches.
2. Select the state where you want to add the action in the diagrammer and click the **New Action** icon in the context panel.
3. In the **New Action** window, click the link for type of action that you want to create. For information about action types, see Workflow Actions Overview.
4. Enter the following basic properties. The properties that appear depend on the type of action.

Property	Description
Workflow	Name of the workflow that includes the state for which you are creating the action. This field is not editable.
State	Name of the state where you want to include the action. This field defaults to the current state. You can use this property to create the action in another state or move the action to another state. For more information, see Moving an Action.
Type	Type of action. This field is not editable.
Script ID	Optional script ID for the action. This value must be lowercase. It cannot include spaces or exceed 26 characters. NetSuite prepends `workflowaction` to the value of this field. If you do not specify a script ID, NetSuite generates a unique ID.

Property	Description
	You can edit the script ID after you create the action. See Editing an Action.
Insert Before	Location in the current list of actions for a state in which to insert the action. Actions for the same trigger type execute in the order in which they appear on the **Actions** subtab for the state. Use this field to reorder actions in a state. See Reordering Actions.
Trigger On	Type of trigger on which to execute this action. This property is required. For more information, see Workflow Triggers and Triggers Reference.
Event Types	The activity that resulted in the record being created, viewed, or updated. The action only executes if the event occurred. For example, you can limit the action to execute only if a record was copied. For more information, see Workflow Event Types and Event Types Reference.
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.
Client Fields	Field or fields that, if changed, will affect another field. Displays record fields that can be edited. This property only appears only for actions that can be executed in the browser. You must select a client trigger for the **Trigger On** property. For more information, see Using Conditional Fields with Actions.
Inactive	Indicates the action does not execute. Use this option to keep an action without deleting it, for example, when you troubleshoot state behavior.

5. Optionally, to enter a condition for the action, use the Condition Builder or Formula Builder to enter a condition.

   The condition criteria must be met for the action to execute. Use a condition to limit the situations in which an action executes, for example, only for a specific user. For more information, see Action Conditions.

6. Optionally, select a saved search as a condition for action execution. The action executes if the current record in the workflow is returned from the saved search.

   To appear in the dropdown list, the saved search record type must be the same as the base record type for the workflow. For more information, see Executing an Action with a Saved Search Condition.

7. Optionally, enter a schedule. The action executes on the schedule you enter after the record enters the state in the workflow. You must select **Scheduled** for the **Trigger On** property. For more information, see Scheduling an Action.

8. Under **Parameters**, enter the properties specific to the action. See Workflow Actions Overview.

9. Click **Save.** The action appears in the context panel under its trigger type category and on the **Actions** subtab for the workflow state.

Editing an Action

Edit an existing action to change action properties and action conditions. Edit an action, for example, during workflow testing and troubleshooting or after you move or copy the action from another state.

To edit an action:

1. If you have not already done so, open the existing workflow that contains the action that you want to edit.

   See Viewing Existing Workflows and Workflow Searches.

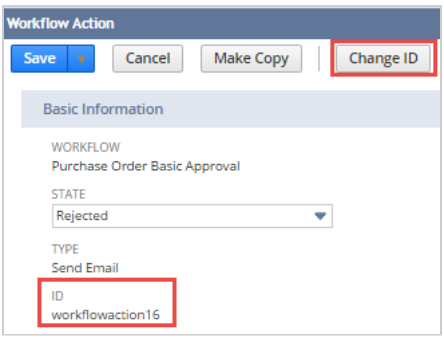
2. Select the state in the diagrammer, hover over the action in the context panel, and click the **Edit** icon.

3. Edit any of the following basic properties. The properties that appear depend on the type of action.

Property	Description
State	Name of the state where you want to include the action. This field defaults to the current state. You can use this property to create the action in another state or move the action to another state. For more information, see Moving an Action.
Script ID	Optional script ID for the action. This value must be lowercase. It cannot include spaces or exceed 26 characters. NetSuite prepends `workflowaction` to the value of this field. If you do not specify a script ID, NetSuite generates a unique ID. You can edit the script ID after you create the action. See Step 5 below.
Insert Before	Location in the current list of actions for a state in which to insert the action. Actions for the same trigger type execute in the order in which they appear on the **Actions** subtab for the state. Use this field to reorder actions in a state. See Reordering Actions.
Trigger On	Type of trigger on which to execute this action. This property is required. For more information, see Workflow Triggers and Triggers Reference.
Event Types	The activity that resulted in the record being created, viewed, or updated. The workflow instance initiates only if the event occurred. For example, you can limit the action to execute only if a record was copied. For more information, see Workflow Event Types and Event Types Reference.
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.
Client Fields	Field or fields that, if changed, will affect another field. Displays record fields that can be edited. This property only appears only for actions that can be executed in the browser. You must select a client trigger for the **Trigger On** property. For more information, see Using Conditional Fields with Actions.

4. To make the state inactive, enable the **Inactive** property. The action does not execute again, even for currently active workflow instances.

5. To change the script ID, click **Change ID**.



In the **Change Script ID** window, enter a new script ID. You can prepend an underscore (_) for readability.

Click **Save**.

6. To edit any condition, see Working with Conditions.
7. To edit the saved search, select a new saved search. To appear in the dropdown list, the saved search record type must be the same as the base record type for the workflow. See Executing an Action with a Saved Search Condition.
8. Under **Parameters**, edit the properties for the type of action. See Workflow Actions Overview.
9. Click **Save**.

Deleting an Action

Delete a workflow action to prevent it from executing on subsequent runs of the workflow. You can use the **Workflow State** window or the context panel to delete an action.

Note: You can also make an action inactive to temporarily prevent it from executing. See Editing an Action.

To delete an action:

1. If you have not already done so, open the existing workflow that contains the action that you want to delete.

   See Viewing Existing Workflows or Workflow Searches.
2. In the diagrammer, select the state that contains the action that you want to delete, hover over the action in the context panel, and click the **Delete** icon.
3. Optionally, select the state in the diagrammer and click the **Edit** icon for the state in the context panel.

   In the **Workflow State** window, click **Edit** next to the action and choose Actions > Delete.

Ordering Actions

In previous releases, new actions were added to the end of a list of actions for a workflow and could be rearranged in any order.

Beginning in 2015.2, a set of rules was created to organize actions in the same way on both the global list and on the context panel. Workflow actions must be ordered based on their triggers, according to the following order:

1. Entry
2. Exit
3. Before Record Load
4. Before User Edit
5. Before Field Edit
6. After Field Edit
7. After Field Sourcing
8. Before User Submit
9. After Record Submit
10. Scheduled

When a user adds a new action to a workflow, the new action is added right after the last action of the same trigger. If the trigger has no other action, SuiteFlow goes up the order of triggers and tries to do the same for each preceding trigger. If no such preceding action is found, the new action is added to the beginning of the list of actions.

When a user changes the trigger of an existing action, the action is placed in the same position as if it were newly added. This position is immediately after the last action of the same trigger or preceding trigger, or at the beginning of the list of actions if no preceding action is found.

When a user wants to move an action, if there are other actions with the same trigger, the action can be placed before or after an action with the same trigger. Alternately, it can be placed right after the last action with the preceding trigger. SuiteFlow goes up the order of triggers and picks the first **last action** available. If there are no actions with preceding triggers, the action can be moved to the first position in the list.

If you attempt to move an action to an inappropriate place in the action list, an error appears telling you where you can move the action within the list of actions.

Note: The global action order can be viewed in the **Workflow State** window. In the context panel, actions are grouped by their trigger type, so only the mutual order of actions of the same trigger type is visible.

Reordering Actions

The order of the workflow actions on the **Actions** subtab of the **Workflow State** window determines the order in which the actions executed for each trigger type. If two actions executed on the same trigger type, the first action listed on the subtab executed before the second, if the condition requirements are met.

Beginning in 2015.2, a set of rules was created to organize actions. This set of rules dictates how you can reorder actions within a state. For more information, see Ordering Actions.

Note: For more information about the order of action execution within a state, see Action Triggers and SuiteFlow Trigger Execution Model

You can use two methods to reorder the actions. You can click the icon to the left of **Edit Actions** subtab of the **Workflow State** window and drag to reorder an action. Optionally, each workflow action includes the **Insert Before** property. You can use this property to reorder an action within a state.

To reorder actions:

1. If you have not already done so, open the existing workflow that contains the actions that you want to reorder.

   See Viewing Existing Workflows or Workflow Searches.
2. In the diagrammer, select the state that contains the actions that you want to reorder and click the **Edit** icon for the state in the context panel.
3. Click the icon to the left of **Edit** for the action on the **Actions** subtab, and drag it to a new location. If you attempt to move an action to an inappropriate place in the action list, an error appears telling you where you can move the action within the list of actions.
4. Optionally, to reorder an action with the **Insert Before** property, click **Edit** next to the action you want to reorder.

   On the **Workflow Action** window, select a new value for the **Insert Before** property, and click **Save**.
5. Click **Save**.

Moving an Action

You can move a workflow action to a different state in the same workflow. Use the **State** property for an action to choose the state where you want to move the action. Move an action, for example, when you troubleshoot or redesign a workflow and want an action to execute in a different state.

Use the following guidelines for moving an action:

- The action retains all the original properties in the new state. It appears at the end of the list on the **Actions** subtab for the new state. Reorder the action to change the position.
- Actions in an action group cannot be moved. You can copy these actions. See Copying an Action.
- You cannot move actions that reference a state field. When you click **Save** after moving an action that references a state field, an error appears. Edit the action and resolve the error before you move it.
- If an action contains a condition, the condition becomes read-only when you move the action. Edit the action in the new state to edit the condition.
- The parameters for a Set Field Value action are set to blank when you move it.

To move an action:

1. If you have not already done so, open the existing workflow that contains the action that you want to move.

   See Viewing Existing Workflows or Workflow Searches.

2. In the diagrammer, select the state that contains the action that you want to move, hover over the action in the context panel, and click the **Edit** icon.
3. On the **Workflow Action** window, select a new value for the **State** property, and click **Save**.
4. Click **Save**.
5. Edit the action in the new state. You can reorder the action or edit the action to make the condition editable. See Editing an Action.

Copying an Action

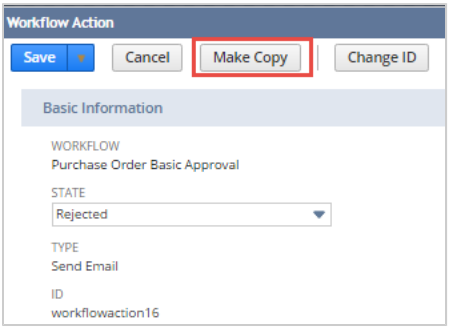
You can copy a workflow action within the same workflow state. After you copy an action, the copy opens in the Workflow Manager. You can then edit the copy of the action.

You can also copy actions that are part of an action group. When you copy an action in an action group, the copy of the action appears at the end of the action group list. For more information about action groups, see Using Action Groups.

To copy an action:

1. If you have not already done so, open the existing workflow that contains the action that you want to move.

   See Viewing Existing Workflows or Workflow Searches.
2. In the diagrammer, select the state that contains the action that you want to copy, hover over the action in the context panel, and click the **Edit** icon.
3. On the **Workflow Action** window, select click **Make Copy**.



   The copy of the action opens in the **Workflow Action** window.
4. Edit the copy of the action and click **Save**. See Editing an Action.

Using Action Groups

Use action groups to execute multiple actions based on the same set of conditions. Action groups make the maintenance of workflows easier because the conditions and other settings of multiple actions in the group are entered and updated at one time on the parent group.

For example, in approval workflows, the Approve and Reject buttons are added to the form by two Add Button actions. If the conditions of these two buttons are the same, an action group lets you place both Add Button actions into the group, and you only have to enter the action condition one time for both actions.

To use action groups, create an action group, define conditions for the action group, and add actions within the group. You can only create action groups for actions that execute on a server trigger, including Entry, Exit, and Scheduled. You cannot use actions that execute on a client trigger.

To execute actions on Items sublist lines, use sublist action groups. For information, see Using Sublist Action Groups.

To create an action group:

1. If you have not already done so, open the existing workflow that contains the state where you want to create an action group in edit mode.

   See Viewing Existing Workflows and Workflow Manager Interface View and Edit Modes.
2. In the diagrammer, select the state and click the **Edit** icon on the **State** tab in the context panel.
3. Click **New Group**. The Workflow Action Group window appears with the **Type** field set to **Group Actions**.
4. In the **Workflow Action Group** window, enter the following properties for the action group:

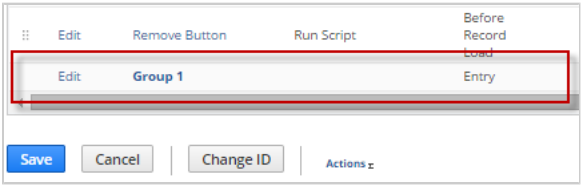
Property	Description
Insert Before	Location in the current list of actions for a state in which to insert the action group. Actions for the same trigger type execute in the order in which they appear on the **Actions** subtab for the state or for a specific action group. Use this field to reorder actions in an action group. See Reordering Actions.
Trigger On	Type of trigger on which to execute this action. This property is required. For more information, see Workflow Triggers and Triggers Reference. **Note:** If you change the trigger type for the action group, NetSuite returns an error if any action in the group is not compatible with the new trigger type.
Event Types	The activity that resulted in the record being created, viewed, or updated. The actions of the action group execute only if the configured event occurs. For example, you can limit the action to execute only if a record was copied. For more information, see Workflow Event Types and Event Types Reference.
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.

5. To make the group inactive, enable the **Inactive** property. The next time the workflow executes, the actions in the group do not execute.
6. To enter a condition for the action group, use the Condition Builder or Formula Builder to enter a condition.

   The condition criteria must be met for the actions in the group to execute. Use a condition to limit the situations in which all actions in a group execute, for example, only for a specific user. For more information, see Working with Conditions.
7. Optionally, select a saved search as a condition for action group execution. The actions in the group execute if the current record in the workflow is returned from the saved search.

   To appear in the dropdown list, the saved search record type must be the same as the base record type for the workflow. For more information, see Executing an Action with a Saved Search Condition.
8. Optionally, enter a schedule. The actions in the group execute on the schedule you enter after the record enters the state in the workflow. For more information, see Scheduling an Action.

9. Click **Save**. The new group appears at the bottom of the **Actions** subtab.



You can also view action groups on the **State** tab in the context panel. For details, see Workflow Context Panel.

10. Click **Edit** to the left of the group to add actions.
11. At the bottom of the **Workflow Action Group** window, click **New Action** to create an action in the action group. For more information, see Creating an Action.

You can also add actions to action groups from the **State** tab in the context panel. Hover over an action group name to view the Add, Edit, and Delete buttons.

Using Sublist Action Groups

Item Sublist Approval in SuiteFlow

You can execute actions on sublist lines as part of workflows for transaction records with Sublist Action Groups. A Sublist Action Group is a group of actions assigned to a workflow state that is to be executed on each line of a sublist. Whether the actions execute during a workflow state is contingent on the group's and group actions' conditions. The sublist action group's condition determine if the group executes. For the group to execute, the group's condition must evaluate to true. If the group's condition evaluates to true, the workflow iterates over all sublist lines and attempts to execute each of the actions within the group. If a group action's condition evaluates to true for a sublist line, the action executes on the sublist line. All actions are evaluated for the possibility of execution on a sublist line before the group moves to the next sublist line. After an action is evaluated, the result of the action is immediately taken into account for the evaluation of subsequent group actions. For example, consider a sublist line's Description field that has a value of Val1. A sublist action group has the two following Set Field Value actions:

- Set Field Value action 1 sets the sublist line's Description field value to Val2.
- Set Field Value action 2 has a condition that will execute the action if the sublist line's Description field's value is Val2.

Both Set Field Value actions are executed. The first action has no condition and changes the value of the sublist line's Description field to Val2. When the second action is evaluated, it sees the value set by the first action and knows to execute next.

You can include the following action types in Sublist Action Groups:

- Set Field Value. This action lets you set the values for record body fields, custom workflow state fields, and sublist fields. You can also set values for these fields depending on the values of the same types of fields (record body, workflow state, sublist).
- Create Record. This action lets you create a new record, depending on the values of record body fields, custom workflow state fields, and sublist fields. For example, when a training item is ordered on a sales order record, you can create a task record for the training supplies for that order.

- Send Email. This action lets you send an email for each item line. You can use all of the line fields, body fields, workflow and state custom fields in the subject and body of the email. To access the currently processing line from within an email template, use the prefix **currentLine**.
- Return User Error. This action returns a user error for the record. You can use all of the line fields, body fields, workflow and state custom fields in the subject and body of the email. Because the action is executed on the server, the error is returned as a full-page message.

Sublist Action Groups can be executed on all server triggers. They cannot be executed on client triggers. For information about workflow triggers, see Triggers Reference.

There are several things to keep in mind about executing sublist action groups and the actions within them on the Before Load Trigger. For details, see Executing Sublist Action Groups on the Before Load Trigger.

Sublist Action Groups are available for all workflow definitions created on transaction records. Sublist Action Groups cannot be used on opportunity records. To configure a sublist action group, you first create the group, then you add actions to the group.

To configure a sublist action group, you first create the group, then you add actions to the group. For information about configuring actions for sublist action groups, see Configuring Actions for Sublist Action Groups.

To create a sublist action group:

1. If you have not already done so, open the existing workflow that contains the state where you want to create an action group in edit mode.

   See Viewing Existing Workflows and Workflow Manager Interface View and Edit Modes.
2. In the diagrammer, select the state and click the **Edit** icon on the **State** tab in the context panel.
3. Click **New Sublist Group**. The Sublist Action Group window appears with the **Type** field set to **Sublist Action Group**.
4. In the **Workflow Action Group** window, enter the following properties for the action group:

Property	Description	Required
Insert Before	Location in the current list of actions for a state in which to insert the action group. Actions for the same trigger type execute in the order in which they appear on the **Actions** subtab for the state or for a specific action group. Use this field to reorder actions in an action group. See Reordering Actions.	No
Trigger On	Type of trigger on which to execute this group. This property is required. Be aware that you cannot execute sublist action groups on client triggers. You can only execute sublist action groups on server triggers. For more information, see Workflow Triggers and Triggers Reference. **Note:** If you change the trigger type for the action group, NetSuite returns an error if any action in the group is not compatible with the new trigger type.	Yes
Event Types	The activity that resulted in the record being created, viewed, or updated. The actions of the action group execute only if the context occurs. For example, you can limit the action to execute only if a record was copied. You can select multiple event types.	No

Property	Description	Required
	For more information, see Workflow Event Types and Event Types Reference.	
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.	No

5. To make the group inactive, enable the **Inactive** property. The next time the workflow executes, the actions in the group do not execute. Inactive sublist action groups are dimmed in the context panel.
6. To enter a condition for the action group, use the Condition Builder or Formula Builder.

   The condition criteria must be met for the actions in the group to execute. Use a condition to limit the situations in which all actions in a group execute, for example, only for a specific user. For more information, see Working with Conditions.

   Be aware that sublist action group's condition can include only record body fields. The actions within the sublist action group can contain both recordy body and sublist fields.
7. Optionally, select a saved search as a condition for action group execution. The actions in the group execute if the current record in the workflow is returned from the saved search.

   To appear in the dropdown list, the saved search record type must be the same as the base record type for the workflow. For more information, see Executing an Action with a Saved Search Condition.
8. In the **Parameters** section of the Sublist Action Group window, the Items sublist is selected in the **Sublist** dropdown list. Currently, you cannot change this selection, as the Items sublist is the only sublist supported for sublist action groups.
9. Optionally, enter a schedule. The actions in the group execute on the schedule you enter after the record enters the state in the workflow. For more information, see Scheduling an Action.
10. Click **Save**. The new group appears in the list of actions for the state on the **Actions** subtab. You can also view sublist action groups on the **State** tab in the context panel. For details, see Workflow Context Panel.

After saving the new group, you can add actions to the group.

To add actions to a sublist action group:

1. In the context panel's **State** tab, point to a sublist action group's name. When you point to a sublist action group, icons appear that let you add actions to the group, edit the group, or delete the group from the state.
2. Click the **+** icon.
3. On the New Action window, click the type of action you want to add to the group. You can select from the following actions:
   - Create Record Action
   - Return User Error Action
   - Send Email Action
   - Set Field Value Action
4. Configure the selected action. See Configuring Actions for Sublist Action Groups.
5. Repeat steps 1 through 4 to add more actions to the group.

Configuring Actions for Sublist Action Groups

Be aware of the following as you configure actions for your sublist action groups:

- Items sublist fields are available for use in conditions, formulas, and action parameters.
- Sublist fields are in alphabetical order in lists of fields. These fields are indicated with the suffix (Line), such as Item (Line).
- To reference a body field in a formula, use the following format: {fieldid}. For example, a body field with an id of memo should be referenced as {memo}.
- To reference a sublist field in a formula, use the following format {line.fieldid}. For example, a sublist field with an id of description should be referenced as {line.description}.
- Body fields in conditions and parameters do not have a prefix or suffix.
- Sublist fields are identified in conditions and parameters with the prefix Line : . For example:
  - In the action list of the sublist action group, in the Parameters column, parameters are listed in the following format: Line : Description = Line : Item : Store Description .
  - In the action list of the sublist action group, in the Condition column, conditions are listed in the following format: Condition : Line : Description = test.
  - On the Set Field Value action detail page, in the Condition field, conditions are listed in the following format: Condition : Line : Description = test.

Executing Sublist Action Groups on the Before Load Trigger

Be aware of the following for sublist action groups and the actions within sublist action groups that are set to trigger on Before Record Load:

- You can get Items sublist field values on new and existing records on the Before Record Load trigger. For example, obtaining a value to evaluate in a condition.
- You can set body fields, Items sublist fields, workflow, and state field values on new and copied records on the Before Record Load trigger.
- You cannot set body or Items sublist fields on existing records on the Before Record Load trigger.
- You can set workflow and state fields on existing records on the Before Record Load trigger.

Scheduling an Action

You can use the Scheduled trigger with an action to create a scheduled action. Scheduled actions only execute after the record enters the state for the action. Use a scheduled action, for example, to send an email at a certain length of time after a record enters a state. For more information about how scheduled actions work, see the Scheduled Trigger.

Important: You can only schedule the Create Record, Initiate Workflow, Send Campaign Email, Send Email, Subscribe To Record, or Custom actions. In addition, these actions only execute if the workflow is in released mode. See Release Status.

To schedule an action:

1. If you have not already done so, create an action. See Creating an Action.

> **Note:** Make sure you set the **Trigger On** property to **Scheduled**.

2. Under **Schedule** on the **Workflow Action** window, enter the following properties:

Property	Description
Delay	Choose **Delay** if you want the action to execute after a certain number of hours or days after the record has entered the state. Number of units of time specified for the **Unit** property until the action executes. For example, if you set the **Unit** property to **Hour** and enter **1** for **Delay**, the action will be performed 1 hour after the record enters the state.
Time of Day	Choose **Time of Day** if you want the action to execute at a specific time of day. When you choose this option, the **Start Time** field becomes required. Execute the action at a specific time of day. Enter the **Start Time**, **Recurrence**, and **Unit**.
Start Time	Time of day the action executes.
Recurrence	Numerical value for the amount of time to pass for the action to be executed again. **Note:** If the condition is not met when the action is scheduled to execute, NetSuite does not attempt to execute the action until the next recurrence.
Unit	Unit of time for the **Delay** or **Time of Day** options. Select **Hour** or **Day**.

3. Click **Save**.

Working with Transitions

A transition moves the record in the workflow from one state to another state. A workflow must have at least two states before you can create a transition. For an overview of transitions in SuiteFlow, including transition properties, triggers, and conditions, see Workflow Transitions.

You can use either the diagrammer or the **Workflow State** window to create and edit transitions. Each transition contains properties that define the state to which the workflow transitions, the trigger on which the transition executes, and any condition that must be met for the transition to execute. In addition, you can also define the transition to execute on the click of a button, on the completion of another state in a workflow, or on a schedule.

The following table describes where you can get more information about working with transitions:

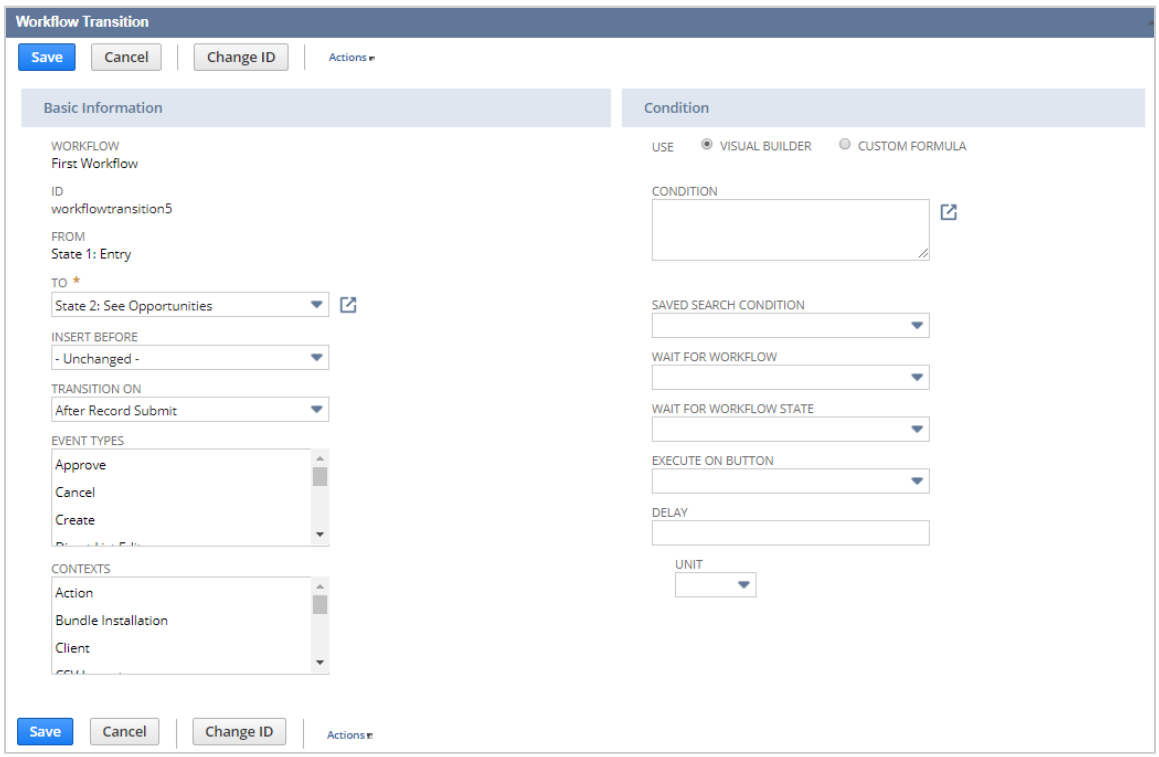
Task	For more information, see ...
Creating a transition in a workflow	Creating a Transition
Editing an existing transaction	Editing a Transition
Deleting a transition and the effects on workflow state properties	Deleting a Transition
Scheduling a transition	Scheduling a Transition
Leaving the transition trigger blank (examples)	Blank Transition Trigger
Executing a transition with a saved search condition (example)	Executing a Transition with a Saved Search Condition

Task	For more information, see ...
Transitioning on completion of a child workflow (example)	Specifying States for Child Workflow Transitions

Creating a Transition

You create a workflow transition in the diagrammer or in the **Workflow State** window for the state that you want to transition from. When you create a transition in the diagrammer, you create the transition and then edit it to specify the properties. When you create a transition from the **Workflow State** window, you specify the properties when you create the transition.

The following screenshot shows the properties for a transition:



Creating a Transition in the Diagrammer

You can create a workflow transition in the Workflow Manager diagrammer. Create the transition and then edit it to specify the transition properties.

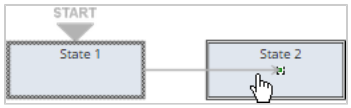
To create a transition in the diagrammer:

1. If you have not already done so, open the existing workflow where you want to create the transition.

   See Viewing Existing Workflows and Workflow Searches.

2. In the diagrammer, hover over the bottom of the state that you want to transition from. The cursor becomes a filled half-circle.

3. Drag the icon to the state to which you want to transition.



4. Release the mouse to create the transition.

5. To edit the transition, click the **Edit** icon on the **Transition** tab in the context panel. See Editing a Transition.

Creating a Transition in the Workflow State Window

You can create a transition in the **Workflow State** window and specify the transition properties.

To create a transition in the Workflow State window:

1. If you have not already done so, open the existing workflow where you want to create the transition.

   See Viewing Existing Workflows and Workflow Searches.
2. Select the state in the diagrammer and click the **Edit** icon on the **State** tab in the context panel.
3. In the **Workflow State** window, click **New Transition** on the **Transitions** subtab.
4. The **Workflow** and **From** fields are already populated with the workflow name and workflow state. You cannot edit these fields.

   Enter the following basic properties:

Property	Description
To	Name of the state to which you want to transition.
Insert Before	Location to insert this transition. Transitions execute in the order in which they appear on the **Transitions** subtab for the state.
Transition On	Optional type of trigger on which to execute this transition. For more information about triggers, see Workflow Triggers and Triggers Reference. For more information about when to leave this property blank, see Blank Transition Trigger.
Event Types	The activity that resulted in the record being created, viewed, or updated. The workflow instance initiates only if the event occurred.

Property	Description
	For example, you can limit the transition to execute only if a record was copied. For more information, see Workflow Event Types and Event Types Reference.
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.

5. To specify conditions for execution of the transition, enter the following properties:

Property	Description
Condition	Criteria that must be met for the transition to execute, which can include a condition or a formula. All specified criteria must be met for the transition to execute. Use a condition to limit the situations in which a transition executes. For more information about using conditions, see Workflow Conditions.
Saved Search	Adds a saved search as a condition for transition execution. The transition executes if the current record in the workflow is returned from the saved search. To appear in the dropdown list, the saved search record type must be the same as the base record type for the workflow. For more information, see Executing an Action with a Saved Search Condition.
Workflow	Execute the transition when the workflow in the **Workflow** dropdown list completes. You can also select a specific state in the **State** dropdown list. The child workflow can be any workflow of the same base record type as the parent workflow. See Specifying States for Child Workflow Transitions.
State	Name of a state in the workflow selected in the **Workflow** dropdown list. When this state completes, the transition executes. See Specifying States for Child Workflow Transitions.
Button	Execute the transition when a user clicks the button on the current record. You must add the button with the Add Button action before you can select it in this dropdown list. See Using Buttons to Execute Transitions.
Delay Unit	Schedule transition to execute on a delayed period of time after the record enters the state. Specify the unit in the **Unit** field and the amount of time in the **Delay** field. You can use these options even if the trigger type is not **Scheduled**. For more information, see Scheduling a Transition.

6. Click **Save**.

Editing a Transition

Edit an existing transition to change transition properties and transition conditions. When you edit a transition, the changes you make only affect future instances of the workflow. The changes do not affect currently running workflow instances. Edit an transition, for example, during workflow testing and troubleshooting or after you create the transition in the diagrammer.

Edit a transition to complete the following tasks:

- Set the properties for the first time if you create the transition by drag and drop in the diagrammer
- Edit any of the other basic properties for the transition, including the trigger, event type, or context
- Edit any of the conditions for transition execution

- Change the target state for the transition

You can also reorder the transitions for a state. The transitions for a state execute in the order that they appear on the **Transitions** subtab for a state. Reordering Transitions.

To edit a transition:

1. If you have not already done so, open the existing workflow that contains the transition that you want to edit.

   See Viewing Existing Workflows and Workflow Searches.
2. In the diagrammer, select the transition and click the **Edit** icon on the **Transition** tab in the context panel
3. Edit any of the following basic properties:

Property	Description
To	Name of the state to which you want to transition.
Insert Before	Location to insert this transition. Transitions execute in the order in which they appear on the **Transitions** subtab for the state.
Transition On	Optional type of trigger on which to execute this transition. For more information about triggers, see Workflow Triggers and Triggers Reference. For more information about when to leave this property blank, see Blank Transition Trigger.
Event Types	The activity that resulted in the record being created, viewed, or updated. The workflow instance initiates only if the event occurred. For example, you can limit the transition to execute only if a record was copied. For more information, see Workflow Event Types and Event Types Reference.
Contexts	The NetSuite functionality or feature used to create, view, or update the record. The workflow instance initiates only if the context occurred. For example, you can limit the workflow instance initiation to only when the record for the workflow was created by a web service. For more information, see the help topics Execution Contexts and Execution Context Types.

4. To edit any condition for execution of the transition, edit the following properties:

Property	Description
Condition	Criteria that must be met for the transition to execute, which can include a condition or formula. All specified criteria must be met for the transition to execute. Use a condition to limit the situations in which a transition executes. For more information about using conditions, see Workflow Conditions.
Saved Search	Adds a saved search as a condition for transition execution. The transition executes if the current record in the workflow is returned from the saved search. To appear in the dropdown list, the saved search record type must be the same as the base record type for the workflow. For more information, see Executing an Action with a Saved Search Condition.
Workflow	Execute the transition when the workflow in the **Workflow** dropdown list completes. You can also select a specific state in the **State** dropdown list. The child workflow can be any workflow of the same base record type as the parent workflow. See Specifying States for Child Workflow Transitions.
State	Name of a state in the workflow selected in the **Workflow** dropdown list. When this state completes, this transition executes.

Property	Description
	See Specifying States for Child Workflow Transitions.
Button	Execute the transition when a user clicks the button on the current record. You must add the button with the Add Button action before you can select it in this dropdown list. See Using Buttons to Execute Transitions.
Delay Unit	Schedule transition to execute on a delayed period of time after the record enters the state. Specify the unit in the **Unit** field and the amount of time in the **Delay** field. You can use these options even if the trigger type is not **Scheduled**. For more information, see Scheduling a Transition.

Deleting a Transition

Delete a workflow transition to prevent it from executing on subsequent runs of the workflow. You can use the diagrammer, context panel, or **Workflow Transition** window to delete a transition.

To delete a transition:

1. If you have not already done so, open the existing workflow that contains the transition you want to delete.

   See Viewing Existing Workflows or Workflow Searches.
2. In the diagrammer, select the transition and either press the Delete key or and click the **Delete** icon on the **Transition** tab in the context panel.
3. Optionally, double-click the transition in the diagrammer. In the **Workflow Transition** window, choose Actions > Delete.

Reordering Transitions

The order of the workflow transitions on the **Transitions** subtab of the **Workflow State** window determines the order in which the transitions execute for each trigger type. If two transitions execute on the same trigger type, the first transition listed on the subtab executes before the second, if the condition requirements are met. Reorder the transitions on the **Transitions** subtab to make the transitions execute in the proper sequence.

Note: For more information about the order of transition execution within a state, see Transition Triggers and SuiteFlow Trigger Execution Model.

You can use two methods to reorder the transitions. You can click the icon to the left of **Edit** on the **Transitions** subtab of the **Workflow State** window and drag to reorder an action. Optionally, each workflow transition includes the **Insert Before** property. You can use this property to reorder a transition within a state.

To reorder transitions:

1. If you have not already done so, open the existing workflow that contains the transitions that you want to reorder.

   See Viewing Existing Workflows or Workflow Searches.
2. In the diagrammer, select the state that contains the transitions that you want to reorder and click the **Edit** icon for the state in the context panel.
3. Click the icon to the left of **Edit** for the action on the **Transitions** subtab, and drag it to a new location.

4. Optionally, to reorder an transition with the **Insert Before** property, click **Edit** next to the transition you want to reorder.

   On the **Workflow Transition** window, select a new value for the **Insert Before** property, and click **Save**.
5. Click **Save**.

Scheduling a Transition

You can use the Scheduled trigger with a transition to create a scheduled transition. You can schedule a transition to execute after a certain number of hours or days have passed after the record enters the state for the transition. For more information about how scheduled transitions work, see the Scheduled Trigger.

You can set the delay for a transition for any trigger type. However, use the Scheduled trigger type to exercise more control over when a transition executes than the other trigger types. For example, if you set the delay on a transition and use the After Record Submit trigger, the delay only starts after the After Record Submit trigger executes. If you use the Scheduled trigger, the transition executes after the delay has passed, starting when the record enters the state in the workflow.

To schedule a transition:

1. If you have not already done so, create a transition. See Creating a Transition.

   Note: Make sure you set the **Trigger On** property to **Scheduled**.
2. In the diagrammer, double-click the state that contains the transitions that you want to reorder and click **Edit** to the left of the transition on the **Transitions** subtab.
3. In the **Workflow Transition** window for the transition, set the following properties:

Property	Description
Delay	Numerical value for the units of time specified for the **Unit** property until the transition executes.
Unit	Unit of time for the **Delay** property. Select **Hour** or **Day**.

4. Click **Save**.

Working with Conditions

You can use conditions on workflow initiation, workflow actions, and workflow transitions. Use conditions to limit the situations in which a workflow initiates or when actions and transitions execute. Use the Condition Builder or the Formula Builder to create conditions with field/value comparisons, expressions, or formulas. For an overview of conditions in SuiteFlow, see Workflow Conditions.

Note: A single workflow initiation, action, or transition condition can contain multiple conditions, for example, a field value comparison, a formula, and a saved search. All conditions must evaluate to true for a workflow instance to initiate or an action or transition to execute.

The following table describes where you can get more information about working with conditions:

Task	For more information, see ...
Using the Condition Builder to create a condition	Defining a Condition with the Condition Builder

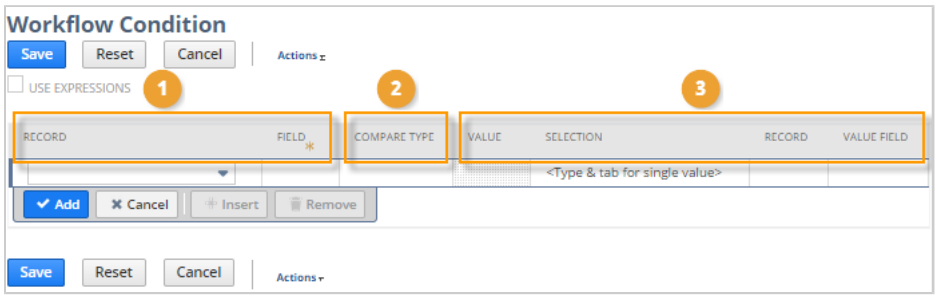
Task	For more information, see ...
Using the Condition Builder to create an expression	Defining a Condition with Expressions
Using the Formula Builder	Defining a Condition with Formulas
Using pre-edit record values in a condition (example)	Referencing Old (Pre-edit) Values in a Workflow
Using conditions with the customer credit hold field (example)	Defining Conditions for Customer Credit Hold Field

Condition Components

Each condition you define with the Condition Builder consists of three separate components:

1. Value to be compared against (left side of the condition). In the Condition Builder, use the left **Record** (optional) and **Field** columns to identify the field to be compared.
2. Compare type. The comparison operator identifies the type of comparison to perform. The types of operators depend on the field values to be compared. Use the **Compare Type** column.
3. Value to be compared against (right side of the condition). In the Condition Builder, use the **Value**, **Selection**, or **Record** and **Value Field** columns to identify the fields to be compared against.

   These three options are mutually exclusive. You cannot combine multiple values to be compared against in a single condition. If you want to use multiple value columns in a single condition, combine the conditions with expressions. See Defining a Condition with Expressions.

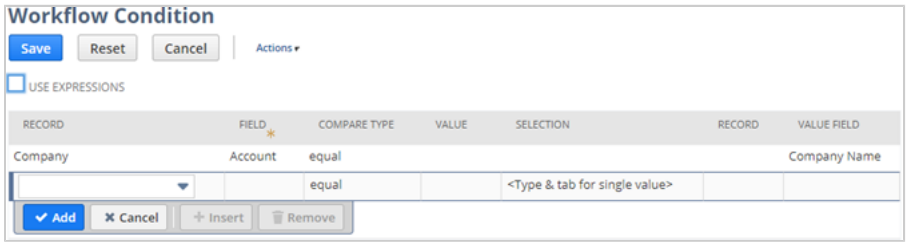


Defining a Condition with the Condition Builder

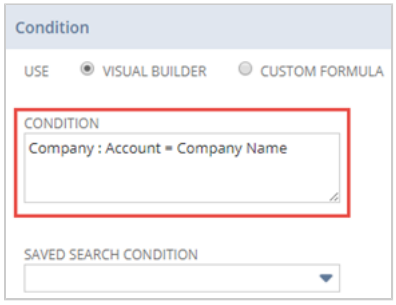
Use the Condition Builder to define a condition using record field value comparisons. Access the Condition Builder from the **Condition** section on the workflow definition, action definition, and transition definition pages.

The following screenshots show a sample condition in the Condition Builder and how it appears on the workflow definition page.

Condition Builder



Action Definition Page

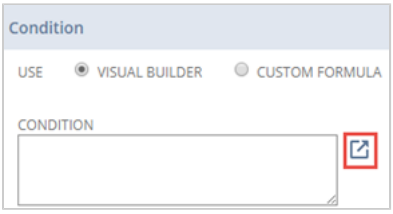


To create a condition with the Condition Builder:

1. If you have not already done so, open the workflow definition page, or action definition, or transition definition page where you want to add the condition.

   See Viewing Existing Workflows or Workflow Searches.

2. Select **Visual Builder** and click the **Open** icon to open the Condition Builder.

   The **Workflow Condition** window opens.

3. Use the following columns to select the value to be compared against:

Column Name	Description
Record	Contains a List/Record type for all records that can be joined with the base record for the workflow. Use this column if you want to include fields from a joined record in the condition. Leave this column blank if you want to use a field from the base record of the workflow. You can also specify the pre-edit record with the **Old Record** option in this column. See Referencing Old (Pre-edit) Values in a Workflow.
Field	Field for the base workflow record or joined record to use in the condition.

> **Note:** On the **Record** list, *Entity* lets you select transaction and expense related fields. *Entity (Employee)* lets you select employee related fields.

4. In the **Compare Type** column, select the comparison operator.

   The available comparison operators depend on the data type of the record field.

5. Select the value to be compared against in the **Value**, **Selection**, or **Record** and **Value Field** columns:

Column Name	Description
Value	Static value for the field. Use the **Selection** column or the **Record** and **Value Field** columns if you want a dynamic value.

Column Name	Description
	If the **Field** column is of type List/Record, you cannot enter a static value. You cannot enter a value for this column for the following compare types: ■ empty ■ not empty ■ checked ■ not checked If you select the Date, Datetime, or Time of Day field types for your condition's record **Field** column: you must enter a valid date in the **Value** column when the **Compare Type** column selection is neither empty nor not empty. If a valid date value is not specified, an alert appears, and the row is not added. The format of the dates you enter for date field types must match your Date Format preference. For information about setting your account preferences, see the help topic General Personal Preferences.
Selection	List/record value for the field that you want to compare to the field value of the **Field** column. If the **Field** column is a List/Record, use the multi-select popup to choose the values. Click in the field and click the **Select Multiple** icon. You cannot enter a value for this field for the following compare types: ■ empty ■ not empty ■ checked ■ not checked
Record	Record that contains the field value that you want to compare to the **Field** column. Contains a List/Record type for all records that can be joined with the base record for the workflow. Use this column if you want to include fields from a joined record in the condition. You can also specify the pre-edit record with the **Old Record** option in this column. See Referencing Old (Pre-edit) Values in a Workflow.
Value Field	Name of the field from the List/Record type of the adjacent **Record** column that you want to compare to the **Field** column. You are required to specify a value for this field when a checkbox field is specified in the **Field** column and a **Compare Type** of equal or not equal is selected. If you do not enter a value for checkbox fields with the equal compare type, an alert appears and the condition row cannot be added. You cannot enter a value for this field for the following compare types: ■ empty ■ not empty ■ checked ■ not checked

6. Click **Add.**

> **Note:** You can also use the **Insert** button to insert the condition into an already existing list of conditions, or select the condition and click **Remove** to delete the currently selected condition.

7. Click **Save** to save the condition and return to the workflow definition, action definition, or transition definition pages.

To modify a line in the Condition Builder after you create it, modify the condition and click **OK**.

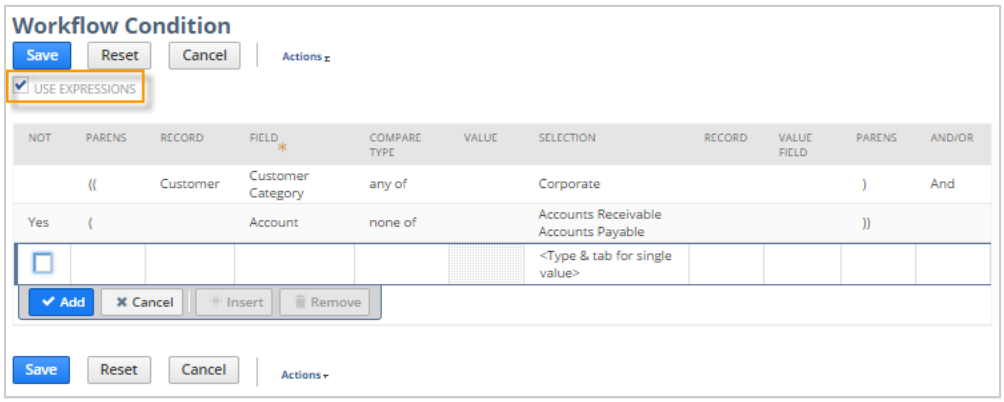
Defining a Condition with Expressions

Use the Condition Builder to define more complicated conditions using record field value comparisons with expressions. Access the Condition Builder from the **Condition** section on the workflow definition, action, and transition pages.

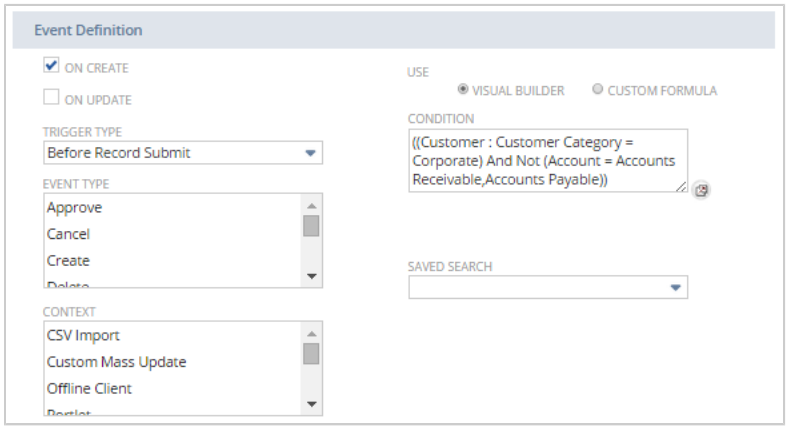
Note: You build conditions with the same process as simpler conditions, except you enable the **Use Expressions** property to use parentheses and NOT, AND and OR operators. See Defining a Condition with the Condition Builder.

The following screenshots show a sample condition with expressions in the Condition Builder and how it appears on the workflow definition page.

Condition Builder



Workflow Definition Page

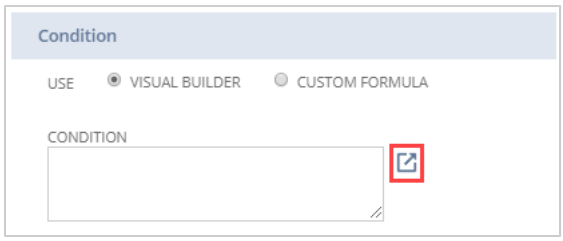


To create a condition with expressions with the Condition Builder:

1. If you have not already done so, open the workflow definition page or action or transition page where you want to add the condition.

   See Viewing Existing Workflows or Workflow Searches.

2. Select **Visual Builder** and click the **Open** icon to open the Condition Builder.



The **Workflow Condition** window opens.

3. Use the **NOT** and **Parens** columns to add the appropriate values:

Column Name	Description
NOT	Indicates the expression is prefixed with NOT. For example, `Not (Total Amount < 50000.00)`.
Parens	Prepend open parenthesis or parentheses to the expression.

4. Use the following columns to select the value to be compared against:

Column Name	Description
Record	Contains a List/Record type for all records that can be joined with the base record for the workflow. Use this column if you want to include fields from a joined record in the condition. Leave this column blank if you want to use a field from the base record of the workflow. You can also specify the pre-edit record with the **Old Record** option in this column. See Referencing Old (Pre-edit) Values in a Workflow.
Field	Field for the base workflow record or joined record to use in the condition.

5. In the **Compare Type** column, select the comparison operator.

   The available comparison operators depend on the data type of the record field.

6. Select the value to be compared against in the **Value**, **Selection**, or **Record** and **Value Field** columns:

Column Name	Description
Value	Static value for the field. Use the **Selection** column or the **Record** and **Value Field** columns if you want a dynamic value. If the **Field** column is of type List/Record, you cannot enter a static value.
Selection	List/record value for the field that you want to compare to the field value of the **Field** column. If the **Field** column is a List/Record, use the multi-select pop-up to choose the values. Click in the field and click the **Select Multiple** icon.
Record	Record that contains the field value that you want to compare to the **Field** column. Contains a List/Record type for all record that can be joined with the base record for the workflow. Use this column if you want to include fields from a joined record in the condition. You can also specify the pre-edit record with the **Old Record** option in this column. See Referencing Old (Pre-edit) Values in a Workflow.
Value Field	Name of the field from the List/Record type of the adjacent **Record** column that you want to compare to the **Field** column.

7. Use the **Parens** and **AND/OR** columns to add the appropriate values:

Column Name	Description
Parens	Append close parenthesis or parentheses to the expression.
And/Or	Append AND or OR to the expression. **Note:** You must add an additional condition or the Condition Builder does not add this operator.

8. Click **Add.**

Note: You can also use the **Insert** button to insert the condition into an existing list of expressions, or select the expression and click **Remove** to delete the currently selected expression.

9. Click **Save** to save the expression and return to the workflow definition, action, or transition definition pages.

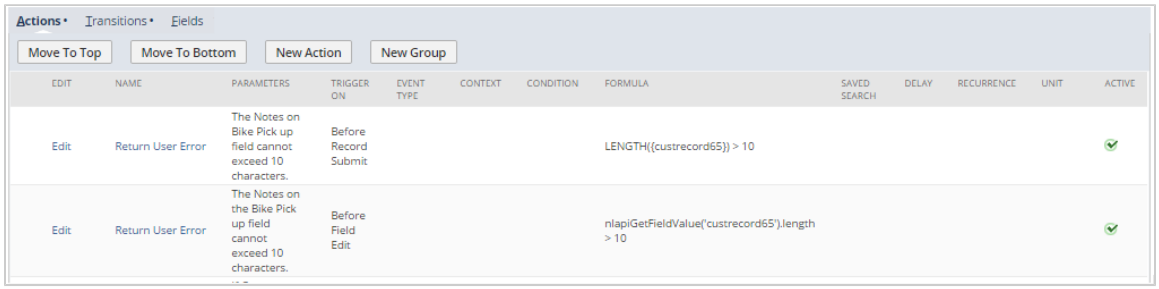
To modify a line in the Condition Builder after you create it, modify the condition and click **OK.**

Defining a Condition with Formulas

Use the Formula Builder to define conditions using formulas. Formula values are dynamically calculated at the time a workflow instance executes. Formula definitions can include field internal IDs to reference fields, SQL functions, JavaScript and SuiteScript, and mathematical operators.

Use the Formula Builder to choose the function to use and the record field to use in the formula. NetSuite generates the formula for you.

The following screenshot shows two formulas as conditions on actions in a workflow:



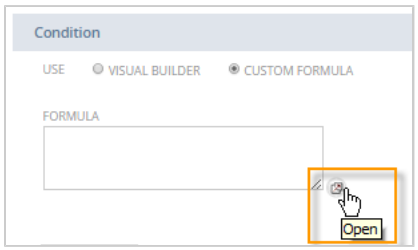
Note: The type of functions you can use depend on the type of trigger for the workflow definition, action, or transition. See Rules and Guidelines for Using Formulas.

To create a formula with the Formula Builder:

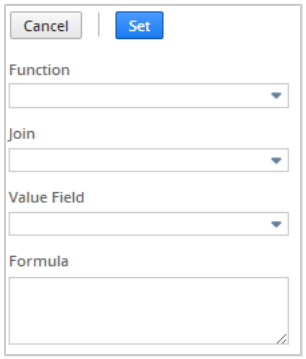
1. If you have not already done so, open the workflow definition page or action or transition page where you want to add the formula.

   See Viewing Existing Workflows or Workflow Searches.

2. Select **Custom Formula** and click the **Open** icon to open the Condition Builder.



The Formula Builder window opens.



Note: Unlike the Visual Builder option, you can type the formula directly into the **Formula** box. Use this option if the function you want to use does not appear for the **Function** property in the Formula Builder.

3. Enter the following properties:

Property	Description
Function	SQL function or JavaScript function to be included in the formula. When you select a function in this dropdown list, it appears in the **Formula** text box. After you select a value, the value disappears. If it does not appear in the **Formula** text box, it is not a valid function. The type of functions that appear depend if the trigger for the workflow definition, action, or transition is a server or client trigger. See Rules and Guidelines for Using Formulas.
Join	Contains List/Record types that can be joined with the base record type for the workflow. You can use this optional field to choose fields in a record other than the base record for the workflow. **Note:** This field does not appear for formulas on actions set to execute on a client trigger.
Value Field	Field with the value that you want to use in the formula. The Formula Builder enters the internal NetSuite ID for the field in the **Formula** text box. NetSuite automatically populates the **Formula** box with the specified function. Override the logic in the box by editing the formula directly.

4. Modify the argument placeholders that appear for the function. Substitute the placeholder with the internal field ID of the field that you want to use in your formula.

   For example, to test if the length is greater than 10, replace the formula:

```
LENGTH(char){custrecord65}
```

with the following formula:

```
LENGTH({custrecord65}) > 10
```

5. Click **Set** to return to the workflow definition, action, or transition page.

Rules and Guidelines for Using Formulas

Important: To avoid errors with formulas, refer to the following rules and guidelines.

- SQL functions execute on the server, while JavaScript and SuiteScript execute on the client. Therefore, use SQL functions with server triggers on workflow definitions, actions, and transitions; use JavaScript and SuiteScript functions with client triggers on actions. You cannot use client triggers with workflow definitions or transitions.
- If a formula references a custom field internal ID and the internal ID changes, make sure you update the formula to use the new internal ID.
- If a formula is invalid, the record or record form does not load properly. Consequently, users cannot enter values on the record form.

Working with Custom Fields

Custom fields include workflow fields and state fields. These fields are variables that you can use in workflows. Workflow fields apply to a single workflow instance. State fields apply to a single state within a workflow.

The following table describes where you can get more information about working with workflow and state fields:

Task	For more information
Creating and using workflow fields	Creating and Using Workflow Fields
Creating and using state fields	Creating and Using State Fields
Storing Workflow Action script return value in a workflow field	Storing a Return Value from a Custom Action Script in a Workflow Field

Creating and Using Workflow Fields

Important: Workflow Custom Fields and Workflow State Custom Fields cannot be successfully used in client-side actions and conditions.

Workflow fields apply to a single workflow definition. The field stores a unique value for each record on which an instance of the workflow runs. You can use a workflow field in conditions and actions like the Set Field Value action.

You can store the value of the field in the database for access by multiple workflow instances for the record type, or only use it temporarily for a single workflow instance.

To create a workflow field, create a workflow field on the workflow definition page. NetSuite populates the field value when the workflow instance runs.

You can also use workflow fields to accomplish the following tasks:

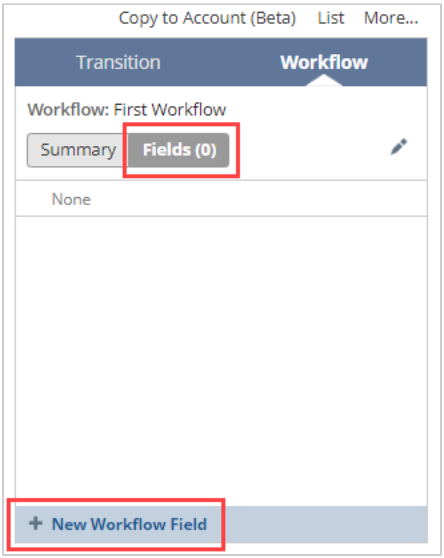
- Track records created by a workflow. Track records created with the Create Record action. Select the workflow field in the **Store Result In** dropdown list on the Create Record action and then use the Subscribe To Record action to track changes to the created record.

  The **List/Record** type of the workflow field must match the **List/Record** type of the record you create with the Create Record action to appear in the **Store Result In** dropdown list.
- Store a return value from a Workflow Action script. Use the **Store Result In** dropdown list of a Custom workflow action to store the value returned from a Workflow Action script. See Storing a Return Value from a Custom Action Script in a Workflow Field.

To create a workflow field:

1. If you have not already done so, open the workflow definition page for the workflow in which you want to create the workflow field.

   See Viewing Existing Workflows or Workflow Searches.
2. In the context panel, click the **Workflow** tab, click the **Fields** view, and then click **New Workflow Field**.



> **Note:** To edit a workflow field after you create it, hover over the field name on the **Fields** view in the context panel and click the **Edit** icon.

3. On the **Workflow Field** window, enter the following properties:

Property	Description
Label	Name or description for the field. NetSuite displays this label in the context panel and in the **Store Result In** dropdown list on Custom and Subscribe To Record actions.
ID	Optional script ID for the field. This value must be lowercase. It cannot include spaces or exceed 30 characters. Use script IDs if you plan to use the SuiteBundler feature to bundle the workflow and install it in another NetSuite account. Script IDs reduce the risk of naming conflicts. NetSuite prepends `customworkflow` to this value when you save the workflow definition.

Property	Description
Owner	Owner of the workflow field.
Description	Optional text description of the field.
Type	Type of custom field. See the help topic Field Type Descriptions for Custom Fields.
List/Record	List or record that corresponds to the list or record type selected for the **Type** property. You must enter a value for this property if you select a **List/Record** or **Multiple Select** value for the **Type** property. See the help topic Creating a Custom Field.
Store Value	Stores the field value in the NetSuite database when the workflow instance runs. Use this option to store the value for future instances of the workflow. If you do not check the **Store Value** box, the field value is **only** accessible in the current workflow instance. When the field value is not stored, the database treats the value as a string and the value is not accessible to other workflow instances.
Workflow	Instances of the workflow to which this field applies. You cannot edit this field.

4. Continue setting the properties on the **Display**, **Validation & Defaulting**, and **Translation** subtabs. The **Sourcing & Filtering** subtab is not available for workflow fields. To source a workflow field, you must use the Set Field Value Action action.

> **Note:** For additional information about workflow field properties, see the help topic Creating a Custom Field.

5. Click **Save**. The workflow field appears on the **Fields** subtab of the workflow definition page and the **Fields** view on the **Workflow** tab on the context panel.

   You can now use the field name in a condition or workflow action.

Creating and Using State Fields

> **Important:** Workflow Custom Fields and Workflow State Custom Fields cannot be successfully used in client-side actions and conditions.

State fields apply to a single state in a workflow. Similar to a workflow field, use a state field to get and set data within a workflow. NetSuite populates the value of the field when the record in the workflow enters the state that contains the state field. You can use the state field in conditions and actions like the Set Field Value action.

The field stores a unique value for each record on which an instance of the workflow runs. You can choose to store the value of the field in the database for access each time a record enters the state, or only use it temporarily for a single visit to the state by the record in the workflow.

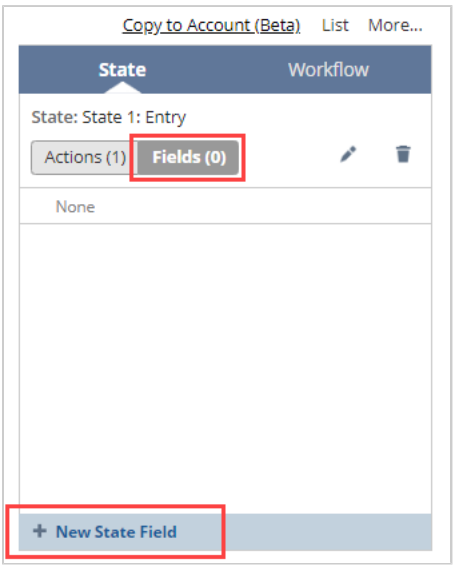
To create a state field, use the **Workflow State** window or the context panel.

To create a state field:

1. If you have not already done so, open the workflow in which you want to create the state field.

   See Viewing Existing Workflows or Workflow Searches.

2. In the diagrammer, select the state in which you want to create the state field, and click **New State Field** on the **Fields** view in the context panel.



To edit an state field after you create it, hover over the field name on the **Fields** view in the context panel, and click the **Edit** icon.

3. On the **Workflow Field** window, enter the following properties:

Property	Description
Label	Name or description for the field. NetSuite displays this label in the context panel and in the **Store Result In** dropdown list on action definition pages.
ID	Optional script ID for the field. This value must be lowercase. It cannot include spaces or exceed 30 characters. Use script IDs if you plan to use the SuiteBundler feature to bundle the workflow and install it in another NetSuite account. Script IDs reduce the risk of naming conflicts. NetSuite prepends `custwfstate` to this value when you save the workflow definition.
Owner	Owner of the state field.
Description	Optional text description of the field.
Type	Type of field. See the help topic Field Type Descriptions for Custom Fields.
List/Record	List or record that corresponds to the list or record type selected for the **Type** property. You must enter a value for this property if you select a **List/Record** or **Multiple Select** value for the **Type** property. See the help topic Creating a Custom Field.
Store Value	Stores the field value in the NetSuite database when the record in the workflow enters the state. Use this option to store the value for future visits to the state by the record in the workflow. If you do not store the field value, the value is only accessible in the for the current visit to the workflow state.
Workflow	Instances of the workflow to which this field applies. You cannot edit this field.

4. Continue setting the properties on the **Display**, **Validation & Defaulting**, **Sourcing & Filtering**, and **Translation** subtabs.

Note: For additional information about custom field properties, see the help topic Creating a Custom Field.

5. Click **Save**. The state field appears on the **Fields** subtab of the **Workflow State** window and the **Fields** view on the **State** tab on the context panel.

   You can now use the field name in a condition or workflow action.

Workflow Administration

Use workflow administration in SuiteFlow to perform searches, initiate, process, and cancel workflow instances, and bundle workflows.

The following table lists the workflow administration tasks:

Task	Description
Search workflows	Search for workflow definitions or workflow instances. See Workflow Searches.
Initiate, process, or cancel workflows	Use the Mass Update feature in NetSuite to initiate workflows, process workflow instances, or cancel workflow instances. See Workflow Mass Updates.
Workflow governance	Refer to workflow governance to view the unit usage limits for workflows and workflow actions. See Custom Actions and Workflow Governance.
Bundle workflows	Use SuiteBundler to bundle a workflow and install it into a different NetSuite account. See Bundling a Workflow.
Specify when workflow instances and history records are kept for a workflow	Specify whether to keep workflow instances and history records Only When Testing, Never, or Always. See Disabling History for a Workflow.
Delete workflow instances and history records	Use the History Record Statistics feature to delete workflow instances and history records. See Deleting Workflow Instances and History Records.

Workflow Instance and History Record Management

Active workflows can generate a large number of workflow instances and history records. The following records are usually generated when a workflow executes:

- An instance record is generated every time a workflow starts a new execution on a record.
- A history record is generated for every state that the workflow enters and exits for each record on which the workflow executes.

There are multiple scenarios that can generate a large number of workflow instances and history records. The following are some of the most common:

- One-state workflows that run every time a record is displayed. (This is the most common cause.)
- Complex workflows that have a lot of states and execute on numerous records.
- You have used NetSuite for a long time and have not deleted any workflow instances or history records.

The accumulation of too many workflow instances and history records can negatively impact NetSuite performance. For example, you may begin to notice slower than usual processing, or unexpected behavior.

You can adjust settings for workflows to avoid the accumulation of too many workflow instances and history records. The following best practices can help you to manage your workflow instances and history records to avoid possible negative performance.

- When you create new workflows, consider disabling workflow history or setting it to only save records during workflow testing. For more information, see Disabling History for a Workflow.

- Regularly delete workflow instances and history records that you do not need to keep. For more information, see Deleting Workflow Instances and History Records.

Disabling History for a Workflow

You can use the Keep Instance and History dropdown list to specify if instance and history records are saved for workflows. Workflow history records take the form of execution and error log records. In the **Keep Instance and History** dropdown list, you can select to save workflow instances and history records **Always**, **Never**, or **Only When Testing**. You can access the Keep Instance and History dropdown list in the following places:

- Workflow Definition Page page
- Editing a Locked Workflow popup window

By default, the Only When Testing option is selected when you create new workflows. Be aware that history records created prior to 2018.1 are saved for existing workflows. You can delete records for these workflows using the Instances and History Records popup window. For details, see Deleting Workflow Instances and History Records.

When you select the Never or Only When Testing options, you can view workflow instance and history records on the record's Active Workflows Subtab and Workflow History Subtab subtabs while the workflow runs. After the workflow finishes or is canceled, the history records related to the workflow instance's execution are deleted from the database. When you select the Always option, history records related to the workflow instance's execution are saved in the database. Be aware that history records are never deleted for workflows that transition to a state with the Do Not Exit Workflow box enabled. For details, see the help topic Exit States.

One of the benefits of not saving history records for workflows that process many records can be improved system performance. Workflows that process many records produce a large amount of history records that can slow system performance. Additionally, a large amount of database space can be freed when you delete workflow history records.

To disable history for a workflow:

1. Go to Customization > Workflow > Workflows.
2. On the Workflows page, click **Edit** next to a workflow for which you want to disable history.
3. In the context panel, on the **Summary** tab, click the **Edit** icon.
4. On the Workflow Definition window, in the **Keep Instance and History** dropdown list, click **Never**.
5. Click **Save**.

You can also disable history for workflows that are locked. See Editing a Locked Workflow.

Deleting Workflow Instances and History Records

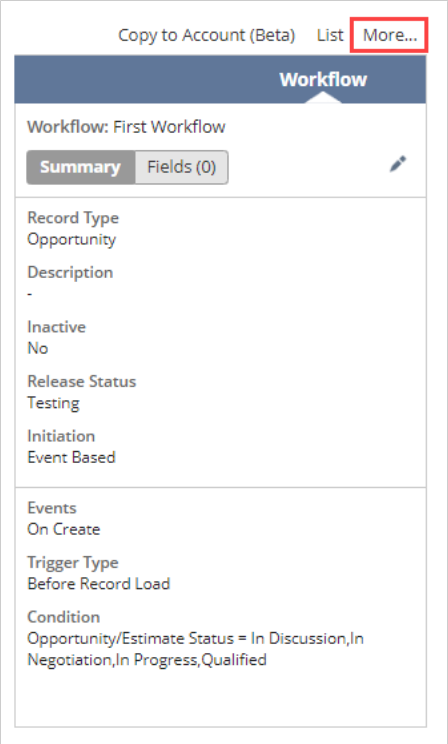
Warning: Deleted workflow instances and history records cannot be recovered.

History records are created each time a workflow moves into a state, including the first state of the workflow. You can delete workflow instances and history records for finished workflow instances and view statistics on the Instances and History Records popup window.

Workflow instances and history records are deleted based on the length of time, in months, they have been stored. 3 months is the default setting, but you can specify a different length of time to base the deletion on.

Note: The number of active and finished workflow instances displayed in the Statistics section on the Instances and History Records popup window is an estimate and not an exact total. The number of active and finished workflow instances are rounded down to the closest number of instances and records.

The following screenshot shows the location of the More menu:



To delete workflow instances and history records:

1. Go to Customization > Workflow > Workflows.
2. Click **Edit** next to the workflow that you want to delete workflow instances and history records for.
3. From the **More** menu, select **Instances and History Records**.
4. In the **Instances and History Records** popup window, in the **Delete** section, enter the number of months for the age of the workflow instances and related history records that you want to delete. To delete all of the workflow instances and related history records, enter **0**.
5. Click **Delete**.

To see the status of the deletion process, from the More menu, select Instances and History Records. Statuses include Scheduling, Waiting for execution, and the progress of the deletion process.

Important: It can take days to delete a large number of history records. Only one deletion process is run at a time. If you initiate multiple deletion processes, they are queued in the order that they were initiated in. The deletion process begins when the preceding deletion process is finished.

A record of the deletion process is logged on the History tab on the Workflow Definition page. When the deletion process is complete, an email is sent to the user who initiated the deletion process.

Workflow Searches

You can use NetSuite to search for workflow definitions and workflow instances. You can either use a basic search or an advanced search for definitions and instances. When you perform a basic search, you use an interface unique to SuiteFlow. When you use an advanced search, you use the standard NetSuite search interface to search for workflow definitions and workflow instances.

Use a workflow search to get information such as which workflows are running, which workflows are inactive, which record types are used in workflows, which employees are using certain workflows, or which workflows came from a bundle installation.

The type of search you use depends on the type of information you want to view:

Type of Search	Type of Search Results	More Information
Workflow definition	For basic searches, search for all workflow definitions that contain properties specified by the search criteria, including workflow name, description, trigger type, and other properties specific to a workflow definition. For advanced searches, you can search for any workflow definition property.	Workflow Definition Search
Workflow instance	For basic searches, search for all workflow instances that contain properties specified by the search criteria, including workflow name, last state entered, date entered and exited workflow, and the date the instance entered a specific state. For advanced searches, you can also search for fields specific to the record for the workflow instance, buttons added with the Add Button action, custom fields, and workflow history fields.	Workflow Instance Search

After you run a workflow search, you can save the search and then add a shortcut to the search results on your dashboard. See Adding Workflow Search Results to the Dashboard.

Note: If you want to search for workflows based on the date the workflow was modified, you should audit the System Notes instead of performing a workflow search. For more information, see the help topic Searching System Notes.

Workflow Definition Search

Use a workflow definition search on the **Workflow Search** page to search all workflows in an account by the criteria defined on the workflow definition page. NetSuite searches all workflows with any release status and also searches inactive workflows.

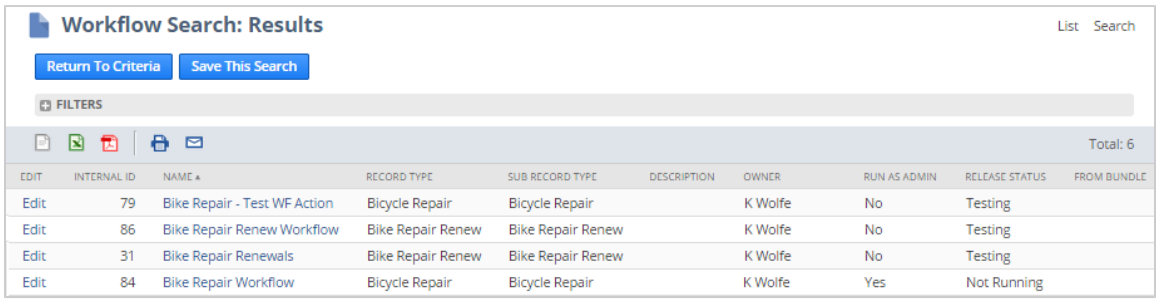
You can access the Workflow Search page from **Search** on the Workflow list page.

You can use the following types of workflow definition search:

- Basic. The basic search is the default search option when you search from the Workflow list page. Use a basic search to perform a search based on a subset of the workflow definition properties. See Performing a Basic Workflow Definition Search.
- Advanced search. Use an advanced search to search on all workflow properties using the standard NetSuite search interface. You can enable advanced search on the **Workflow Search** page. See Performing an Advanced Workflow Definition Search.

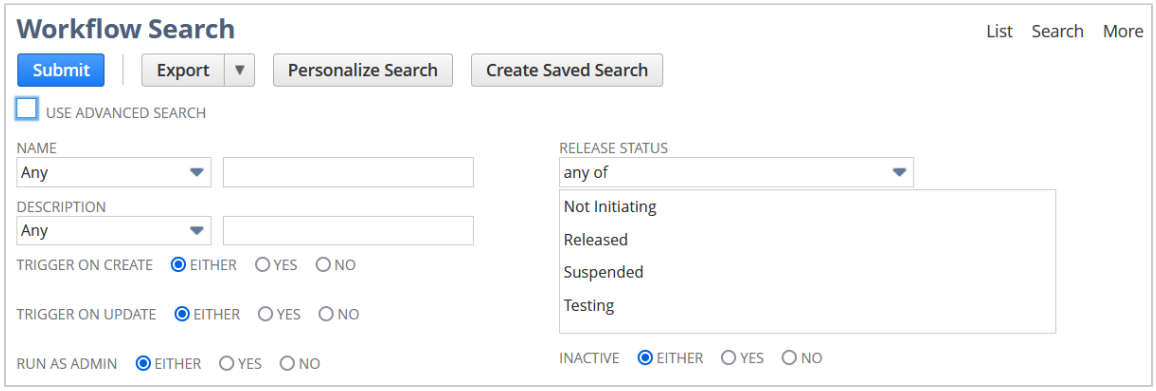
Optionally, you can export the results of a search as an Excel file or save the search as a saved search. You can also click **Personalize Search** to personalize the search form and only include the filters that a specific search requires. For more information, see the help topic Personalizing a Search Form.

The following screenshot shows the results of a workflow definition search:



Performing a Basic Workflow Definition Search

Use basic search to perform a search based on a subset of the workflow properties. The following screenshot shows the workflow properties that you can use as criteria in a basic search:



To perform a basic workflow definition search:

1. Go to Reports > New Search. On the Search page, click **Workflow**.
2. On the Workflow Search page, clear the **Use Advanced Search** box.
3. Enter the following search criteria. For the **Name**, **Description**, and **Release Status** properties, you must also select a comparison operator.

Critera	Description
Name	The **Name** property on any workflow definition. Search by whole words only.
Description	The **Description** property on any workflow definitions. Search by whole words only.
Trigger On Create	Searches for workflows with the **On Create** event type selected. See Initiating a Workflow on an Event.
Trigger On Update	Searches for workflows with the **On Update** event type selected. See Initiating a Workflow on an Event.
Run As Admin	Searches for workflows with the **Execute As Admin** property enabled. See Execute As Admin.
Release Status	Searches for workflows by the **Release Status**. See Release Status.

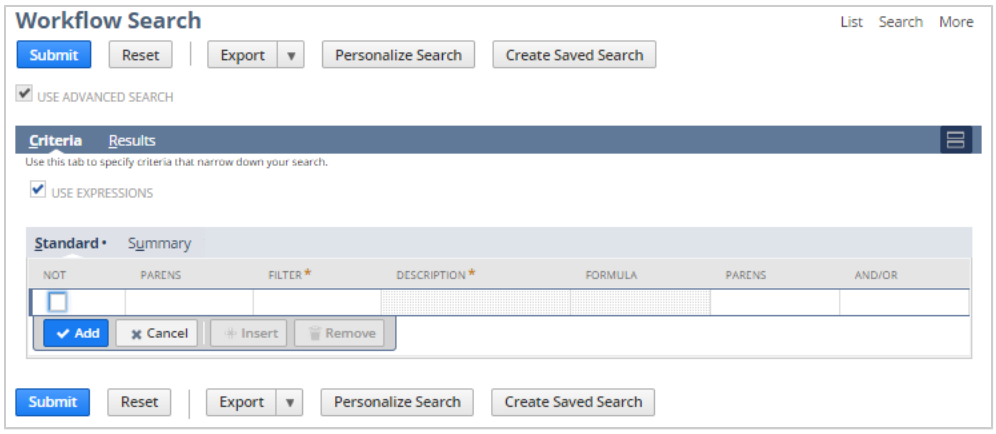
Critera	Description
Inactive	Searches for workflows with the **Inactive** property enabled. See Inactivating a Workflow.

4. (Optional) To customize the results page, click **Personalize Search**. See the help topic Personalizing a Search Form.
5. (Optional) To create a saved search from the search criteria entered on the page, click **Create Saved Search**. NetSuite opens the search criteria on the **Saved Workflow Search** page. For more information about working with saved searches, see the help topic Defining a Saved Search.
6. Do one of the following:
   - To perform the results and display the results in NetSuite, click **Submit**. NetSuite displays the search results.
   - To perform the search and then export the results and view them as a comma-separated file, click Export > Export Data Only.

Performing an Advanced Workflow Definition Search

Use advanced search to perform a search based on all workflow properties available on the workflow definition page. An advanced search uses the standard NetSuite search interface to search for workflows.

The following screenshot shows the advanced workflow search interface:



Note: For more general information about running searches in NetSuite, see the help topic Running Searches.

To perform an advanced workflow definition search:

1. Go to Reports > New Search. On the Search page, click **Workflow**.
2. On the **Workflow Instance Search** page, check the **Use Advanced Search** box.
3. Optionally, on the **Criteria** tab, enable **Use Expressions** to use NOT, AND, OR, and parentheses in the search filters.
4. On the **Criteria** tab, define the standard and summary search filters:

Subtab	Description
Standard	Select results based on the value of individual workflow property values. For more information, see the help topic Defining Standard Search Filters.

Subtab	Description
Summary	Select results based on calculated summary property values for a group of workflows. This filter is available only if you defined a summary type for a field on the **Results** tab. For more information, see the help topic Summary Search Filters.

5. On the **Results** tab, set the sort values for the search results and the search results fields to display, define summary types for results fields, and additional search result options.

   For more information, see the help topic Search Results Display Options.

6. Optionally, to customize the results page, click **Personalize Search**. See the help topic Personalizing a Search Form.

7. Optionally, to create a saved search from the search criteria entered on the page, click **Create Saved Search**.

   NetSuite opens the search criteria on the **Saved Workflow Search** page. For more information about working with saved searches, see the help topic Defining a Saved Search.

8. To perform the results and display the results in NetSuite, click **Submit**. NetSuite displays the search results. You can then export the results or edit the workflow definition.

   -OR-

   To perform the search and then export the results and view them as a comma-separated file, click Export > Export Data Only.

Workflow Instance Search

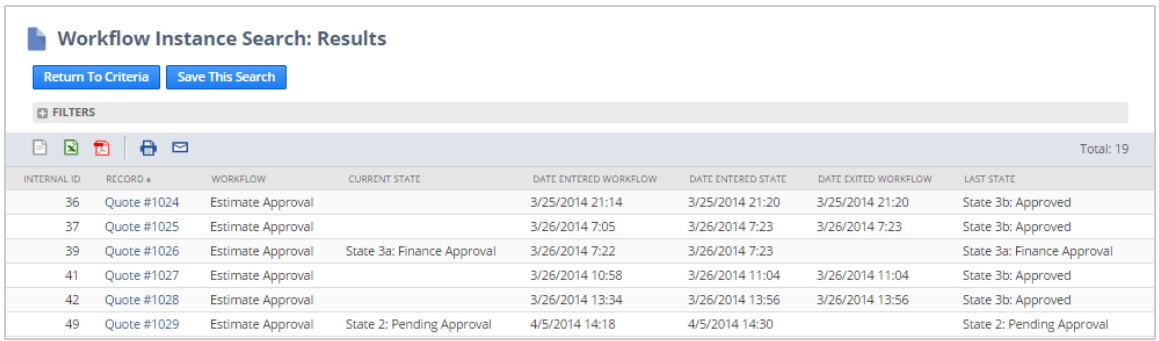
Use a workflow instance search to get a list of workflow instances that completed or are currently in progress for a specific workflow definition or multiple workflow definitions. To obtain the list, run a search with the criteria for the workflow instances that you want to view. After you run the search, you can view the individual records on which each workflow instance runs.

You can use the following types of workflow instance search:

- Basic. The basic search is the default search option when you search for a workflow instance. Use a basic search to perform a search based on a subset of the workflow instance properties. See Performing a Basic Workflow Instance Search.
- Advanced. Use an advanced search to search on all workflow instance properties using the standard NetSuite search interface. You can enable advanced search on the **Workflow Instance Search** page. See Performing an Advanced Workflow Instance Search.

Optionally, you can export the results of a search as an Excel file or save the search as a saved search. You can also click **Personalize Search** to personalize the search form and only include the filters that a specific search requires. For more information, see the help topic Personalizing a Search Form.

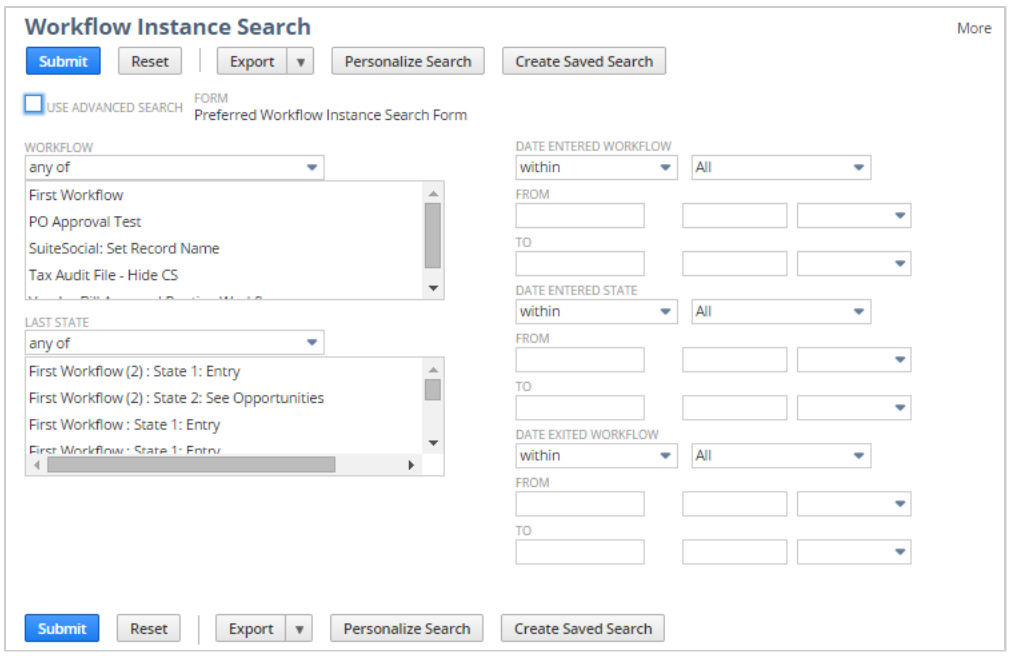
The following screenshot shows the results of a workflow instance search:



Note: The **Current State** column displays states for currently running workflows. If a workflow is not currently running, it does not display a current state.

Performing a Basic Workflow Instance Search

Use basic workflow instance search to perform a search based on a subset of the workflow instance properties. The following screenshot shows the workflow instance properties that you can use as criteria in a basic search:



To perform a basic workflow instance search:

1. Go to Reports > New Search.
2. On the **Search** page, click **Workflow Instance**.
3. On the **Workflow Instance Search** page, make sure **Use Advanced Search** is unchecked and enter the following search criteria.

Note: For the Workflow or Last State criteria, select **any of** or **none of** and then select the names to include in the search or exclude from the search.

For **Date Entered Workflow**, **Date Entered State**, and **Date Exited Workflow**, use the dropdown lists, Quick Filter, and calendar to select the date or date range. See Using Quick Filters For Date Range Selection.

Criteria	Description
Workflow	Include or exclude any workflow currently in the NetSuite account. This list shows all workflows, including inactive workflows.
Last State	The last state entered for all instances of workflows in the **Workflow** list. For completed workflow instances, this list shows the exit state for the workflow instance.

Criteria	Description
	The list shows the workflow name and then the last state name: `<workflow name> : <state name>` Select **any of** or **none of** and then select the state names to include in the search or exclude from the search.
Date Entered Workflow	The date that the workflow instance initiated. Use the dropdown lists, Quick Filter, and calendar to select the date or date range.
Date Entered State	The date that the record in the workflow entered the most recent state in the workflow instance. For completed workflows, this is the exit state. For currently running workflows, this is the last state entered by the record in the workflow instance.
Date Exited Workflow	For completed workflow instances, the date that the record left the exit state. Use the dropdown lists, Quick Filter, and calendar to select the date or date range.

4. Optionally, to customize the results page, click **Personalize Search**. See the help topic Personalizing a Search Form.
5. Optionally, to create a saved search from the search criteria entered on the page, click **Create Saved Search**.

   NetSuite opens the search criteria on the **Saved Workflow Search** page. For more information about working with saved searches, see the help topic Defining a Saved Search.
6. To perform the results and display the results in NetSuite, click **Submit**. NetSuite displays the search results. You can click **Record** to view the individual record or export the search results.

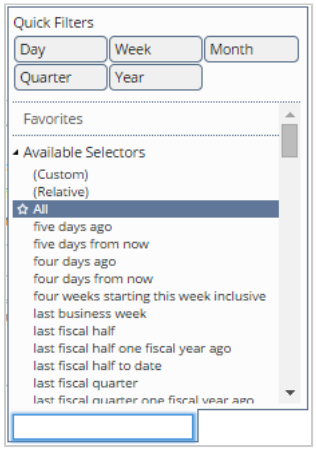
   -OR-

   To perform the search and then export the results and view them as a comma-separated file, click Export > Export Data Only.

Using Quick Filters For Date Range Selection

When you select a date range for search criteria in a workflow instance search, use **Quick Filters** to select a date range and auto-populate the **To** and **From** range fields on the search form or add fields to the date range.

The following screenshot shows **Quick Filters** for a date range:



Use the following features to select a date range:

- Click and highlight one or more of the **Day**, **Week**, **Month**, **Quarter**, and **Year** filters.

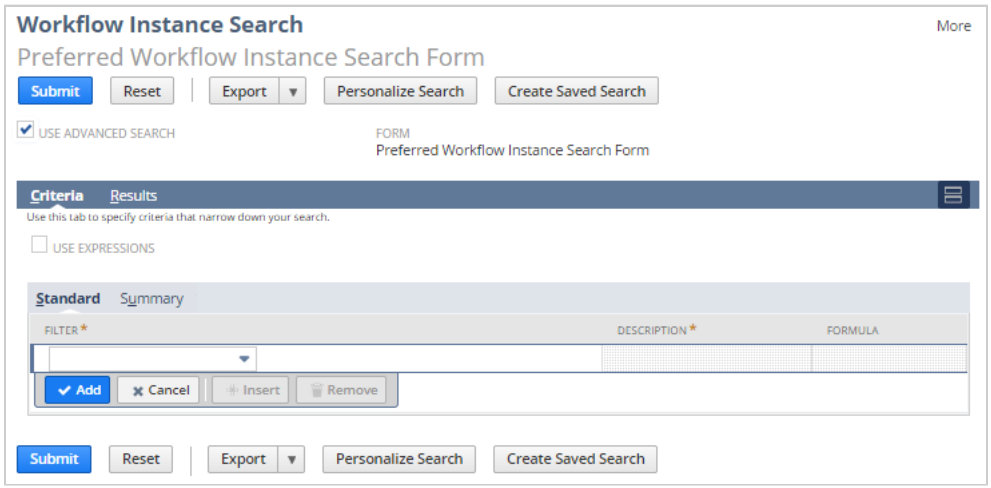
NetSuite filters the Available Selectors list to include only the selectors for the filters you select. If you select multiple filters, the Available Filters list shows selectors that apply to all of the filters.

- Click the star icon next to any selector to add the selector to the Favorites list.
- Select **Custom** to add time fields to the search criteria.
- Select **Relative** to include items from a specific number of seconds, hours, days, quarters, or years in the past or the future as search criteria.

Performing an Advanced Workflow Instance Search

Use an advanced search to perform a search based on any workflow instance property, including fields specific to the record for the workflow instance, buttons added with the Add Button action, custom fields, and workflow history fields. An advanced search uses the standard NetSuite search interface to search for workflows.

The following screenshot shows the advanced workflow instance search interface:



To perform an advanced workflow instance search:

1. Go to Reports > New Search.
2. On the **Search** page, click **Workflow Instance**.
3. On the **Workflow Instance Search** page, check the **Use Advanced Search** box.
4. Optionally, on the **Criteria** tab, enable **Use Expressions** to use NOT, AND, OR, and parentheses in the search filters.
5. On the **Criteria** tab, define the standard and summary search filters:

Subtab	Description
Standard	Select results based on the value of individual workflow instance values. For more information, see the help topic Defining Standard Search Filters.
Summary	Select results based on calculated summary property values for a group of workflow instances; this filter is available only if you defined a summary type for a field on the **Results** tab. For more information, see the help topic Summary Search Filters.

6. On the **Results** tab, set the sort values for the search results and the search results fields to display, define summary types for results fields, and additional search result options.

   For more information, see the help topic Search Results Display Options.

7. Optionally, to customize the results page, click **Personalize Search**. See the help topic Personalizing a Search Form.
8. Optionally, to create a saved search from the search criteria entered on the page, click **Create Saved Search**.

   NetSuite opens the search criteria on the **Saved Workflow Search** page. For more information about working with saved searches, see the help topic Defining a Saved Search.
9. To perform the results and display the results in NetSuite, click **Submit**. NetSuite displays the search results. Click **Record** to view the individual record or export the search results.

   -OR-

   To perform the search and then export the results and view them as a comma-separated file, click Export > Export Data Only.

Adding Workflow Search Results to the Dashboard

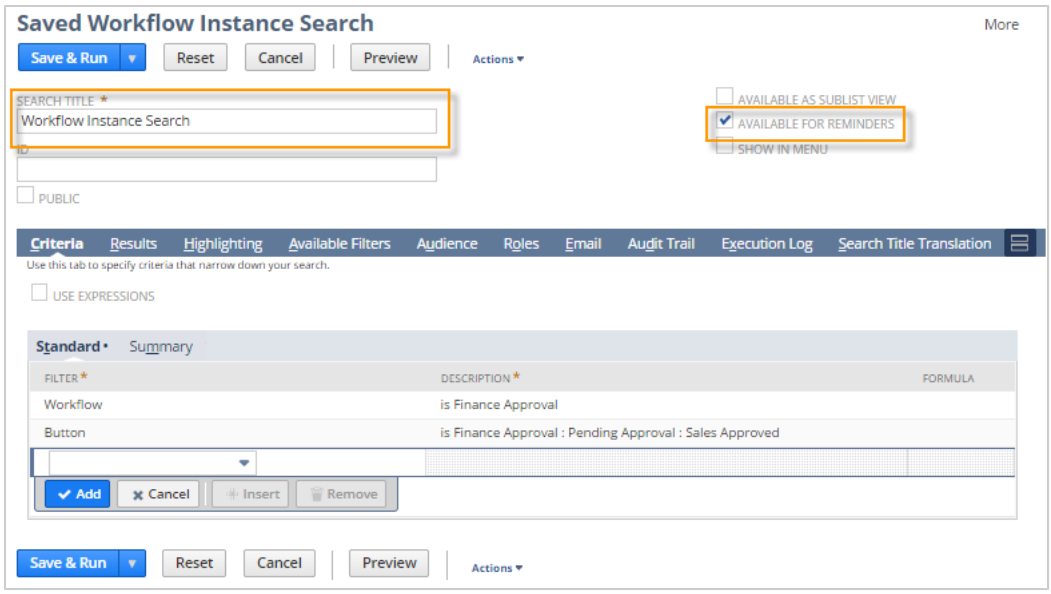
To access the results of workflow definition and workflow instance searches from the dashboard, add shortcut links to custom search portlets or the Reminders portlet. Add search results to the dashboard for common tasks, such as approvals of expense reports, budgets, and purchase requests.

For example, add the results of a workflow instance search that uses the Button filter to the Reminders portlet to show all records that a user must act on for a workflow to continue. For more information about creating a search with the Button filter, see Using the Button Filter in a Workflow Instance Search.

To add workflow search results to the Reminders portlet:

1. Run a workflow definition or workflow instance search. See Workflow Definition Search or Workflow Instance Search.
2. On the search results page, click **Save This Search**.

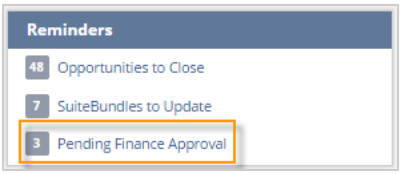
   NetSuite opens the search criteria in **Saved Workflow Search** page:



3. Enter a value for the **Search Title** and enable **Available For Reminders**.
4. Click **Save**.
5. On the dashboard, click **Set Up** on the **Reminders** portlet.

6. In the **Set Up Reminders** window, click **Add Custom Reminders**. Select the saved search you created in steps 3 and 4.
7. Click **Done** and then click **Save**.

   The link to the saved search appears on the **Reminders** portlet:



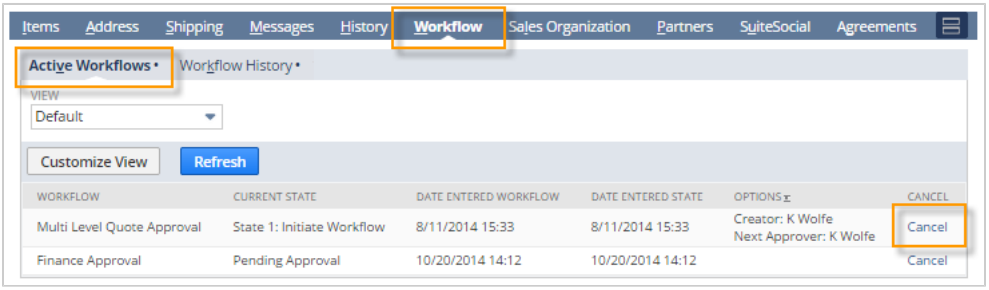
Canceling a Single Workflow Instance

You can cancel a single workflow instance from the record on which the workflow instance runs. Users with the Administrator role can cancel workflow instances. You can also cancel multiple workflow instances simultaneously. See Mass Canceling Workflow Instances.

Important: When you cancel a workflow instance for a single record, NetSuite reloads the record in NetSuite. Therefore, when a workflow is set to initiate **On Update** and uses the All or Before Record Load server triggers, the workflow instance re-initiates when you cancel it. To avoid this result, use the Before Record Submit or After Record Submit server triggers on workflows set to initiate **On Update**. Mass canceling workflow instances does not have this limitation.

To cancel a single workflow instance:

1. Edit or view the record on which the workflow instance runs.
2. Click the **Workflow** subtab. The **Active Workflows** subtab appears by default.
3. Click **Cancel** next to the workflow instance that you want to cancel.



Note: Any other workflow instances still run. You must cancel each workflow instance individually.

4. Click **Yes** in the popup that appears.
5. Optionally, click the **Workflow History** subtab to see the canceled instance. The **Notes** column displays the name of the user who canceled the instance.

Workflow Mass Updates

Use the NetSuite Mass Update feature to work with multiple workflow instances simultaneously. To use the Mass Updates feature with workflow instances, you create a mass update that identifies the

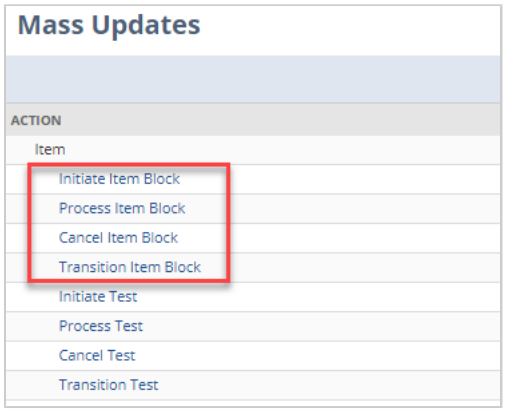
records on which you want to perform the mass update and then save the update or perform the update immediately. The record type of the records on which the mass update is performed matches the base record type of the selected workflow.

Use the Mass Updates feature to perform the following tasks with workflow instances:

- Initiate instances of the workflow. See Mass Initiating Workflow Instances.
- Execute transitions. See Mass Processing Records in a Workflow.
- Cancel workflow instances. See Mass Canceling Workflow Instances.
- Transition an active workflow on a record outside of the defined workflow execution path. See Mass Transitioning Workflow Instances.

For each record type, the Mass Updates page lists the workflows that you can initiate, process, cancel, or transition. For example, the following screenshot shows that there are two workflows based on the Item record:

- Item Block
- Test



Note: For general information about the Mass Update feature in NetSuite, see the help topic Mass Changes or Updates.

Mass Initiating Workflow Instances

You can initiate multiple instances of a specific workflow simultaneously on multiple records in NetSuite. You can mass initiate workflow instances for standard NetSuite record types or custom record types. Use this feature to avoid viewing each record individually to initiate a workflow instance on the record.

For example, use this feature if your company recently imported a large number of employee records for an acquisition. After they are imported, you decide that you want to initiate a workflow named **New Employee Process** on each employee record. NetSuite can initiate a separate workflow instance for each employee record, based on filters you specify for the mass update.

Note: You can also use a scheduled workflow to initiate multiple instances of a specific workflow simultaneously on multiple records. See Scheduling a Workflow.

To mass initiate a workflow on multiple records:

1. Go to Lists > Mass Update > Mass Updates.

2. On the Mass Updates page, expand **Workflow**.

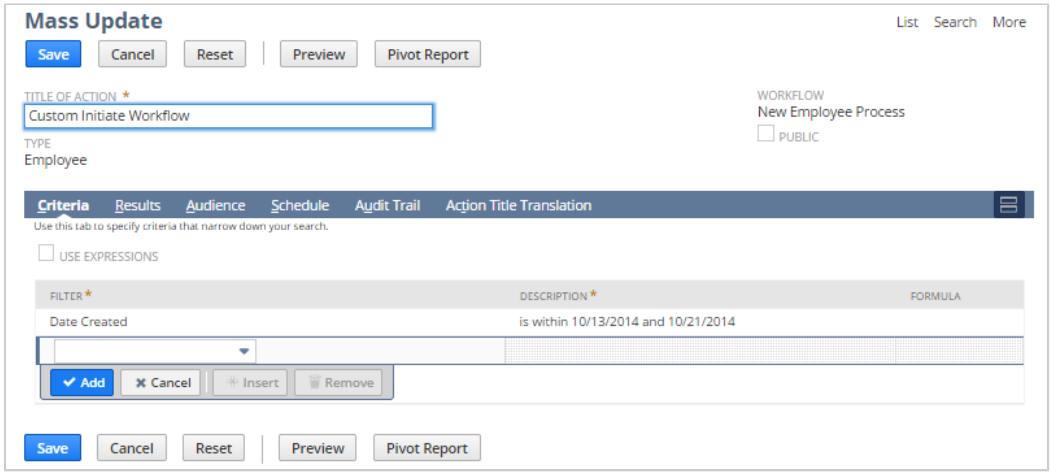
   For each record type in NetSuite, the Mass Updates page lists the workflows that are available to initiate, process, cancel, or transition.

3. Click **Initiate <workflow name>** to initiate the appropriate workflow. The **Mass Update** page displays.

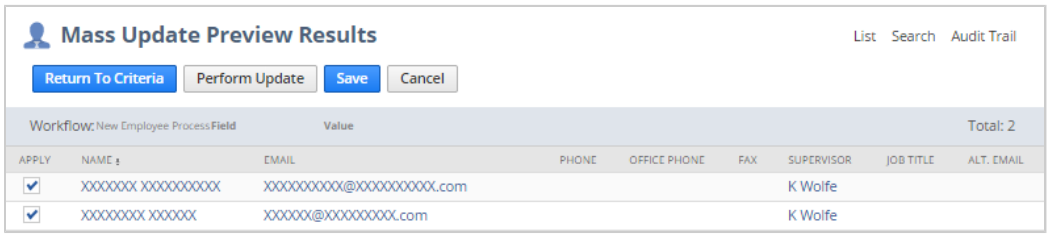
4. On the **Mass Updates** page, enter a **Title** and define the filters and other options on the following subtabs:

Subtab	Required / Optional	Description
Criteria	Required	Select a field or formula in the dropdown list and enter values in the popup that appears. See the help topic Advanced Search Criteria Filters.
Results	Optional	Define display options for mass update results. The results appear when you preview the records for the mass update. See the help topic Search Results Display Options.
Audience	Optional	Define the users who can run the mass update.
Schedule	Optional	Define a schedule on which to perform the mass update. See the help topic Scheduling a Mass Update.

The following screenshot shows the **Mass Updates** page for employee records created between two dates:



5. Click **Preview** to view the records on which the mass update will run:



6. Do one of the following:

   - To perform the update immediately, in the **Apply** column, select the records you want to initiate the workflow on and click **Perform Update**. The status of the mass update is displayed on the **Mass Update Status** page.

- To save the update and perform it later, click **Save**. NetSuite saves the mass update and displays the **Saved Mass Updates** page. You can return to this page to initiate the workflow instances by going to Lists > Mass Update > Saved Mass Updates. Click **Preview** next to the name of the update. On the **Mass Update Preview** page, click **Perform Update** to initiate the workflow instances.

Note: Optionally, from the **Mass Update** page, you can use the **Audit Trail** tab to view the changes to the current mass update or use the **Action Title Translation** tab to localize the title of the mass update.

Mass Processing Records in a Workflow

You can process multiple instances of a specific workflow simultaneously for the same record type in NetSuite. Mass process instances of a workflow to execute a transition from the current state of the workflow instance to the next state. You can use this feature to reevaluate transitions or execute a specific transition that executes on a button. This feature only applies to records that are already in a running workflow instance.

For example, you want to transition all employee records in a workflow named **New Employee Process** from **State 2 Approve Training** to **State 3 Training Approved**. The workflow contains an Approve Training button, added with the Add Button action, that approves the records and transitions them to State 3. Instead of opening each employee record and approving the training, you can mimic the click of the Approve Training button to approve all or a subset of the records and transition them to State 3.

To mass process a workflow for multiple records:

1. Go to Lists > Mass Update > Mass Updates.
2. On the Mass Updates page, expand **Workflow**.

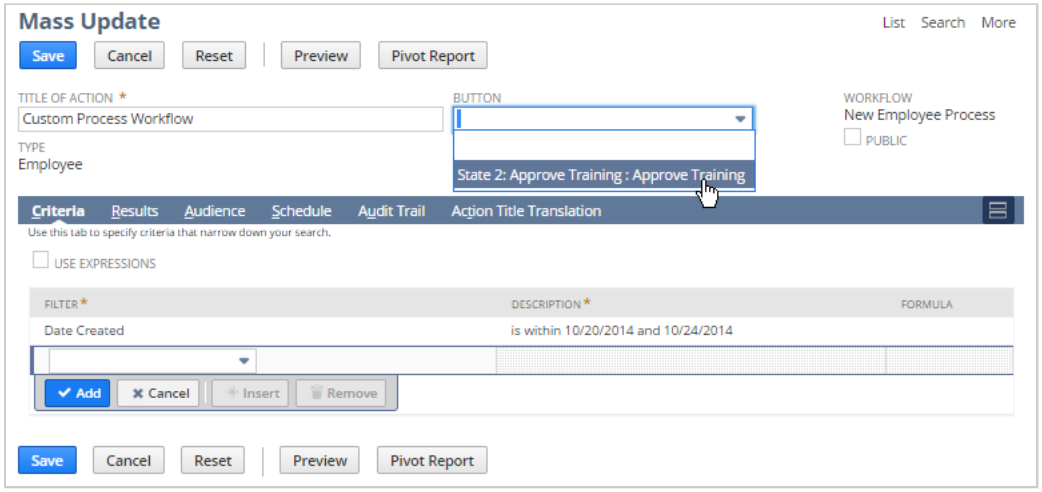
   For each record type in NetSuite, the Mass Updates page lists the workflows that are available to initiate, process, cancel, or transition for each record type.
3. Click **Process <workflow name>** to process the appropriate workflow. The **Mass Update** page displays.
4. On the **Mass Updates** page, enter a **Title**.
5. If you want to mimic the click of a button to transition the record to the next state, select the button name in the **Button** dropdown list.

   Note: This must be a button added with the Add Button action to the current state of the workflow instance.
6. Define the filters and other options on the following subtabs:

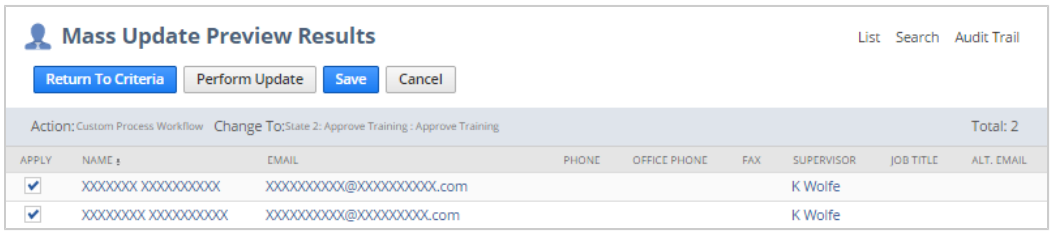
Subtab	Required / Optional	Description
Criteria	Required	Select a field or formula in the dropdown list and enter values in the popup that appears. See the help topic Advanced Search Criteria Filters.
Results	Optional	Define display options for mass update results. The results appear when you preview the records for the mass update. See the help topic Search Results Display Options.
Audience	Optional	Define the users who can perform the mass update.
Schedule	Optional	Define a schedule on which to perform the update.

Subtab	Required / Optional	Description
		See the help topic Scheduling a Mass Update.

The following screenshot shows the **Mass Updates** page for employee records created between two dates:



7. Click **Preview** to view the records on which the mass update will run:



8. Do one of the following:
   - To perform the update immediately, in the **Apply** column, select the records you want to transition and click **Perform Update**. The status of the mass update is displayed on the **Mass Update Status** page.
   - To save the update and perform it later, click **Save**. NetSuite saves the mass update and displays the **Saved Mass Updates** page. You can return to this page to transition the records by going to Lists > Mass Update > Saved Mass Updates. Click **Preview** next to the name of the update. On the **Mass Update Preview** page, click **Perform Update** to transition the records.

Note: Optionally, from the **Mass Update** page, you can use the **Audit Trail** tab to view the changes to the current mass update or use the **Action Title Translation** tab to localize the title of the mass update.

Mass Canceling Workflow Instances

You can cancel multiple instances of a specific workflow simultaneously for the same base record type in NetSuite. You can cancel workflow instances for standard NetSuite record types or custom record types. Use this feature to avoid viewing each record individually to cancel a workflow instance on the record. Users with the Administrator role can cancel workflow instances.

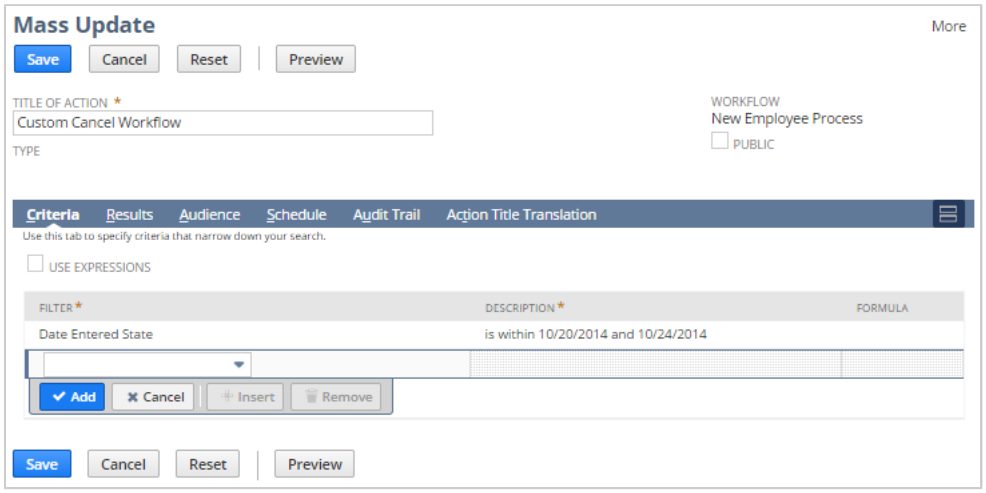
To cancel a workflow instance on multiple records simultaneously:

1. Go to Lists > Mass Update > Mass Updates.
2. On the Mass Updates page, expand **Workflow**.

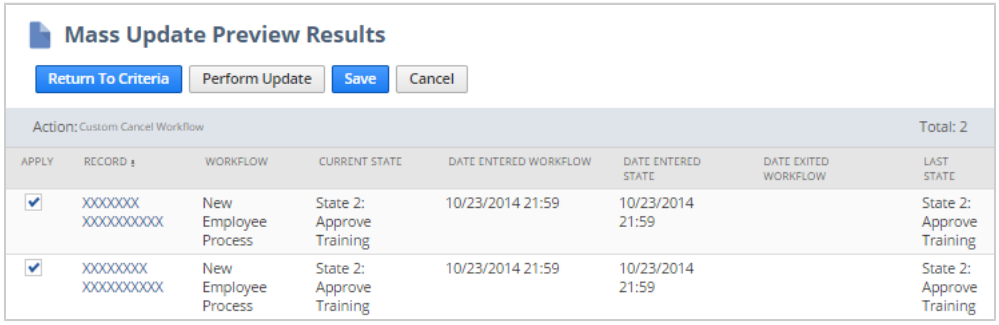
   For each record type in NetSuite, the Mass Updates page lists the workflows that are available to initiate, process, cancel, or transition for each record type.
3. Click **Cancel <workflow name>** to process the appropriate workflow. The **Mass Update** page displays.
4. On the **Mass Updates** page, enter a **Title** and define the filters and other options on the following subtabs:

Subtab	Required / Optional	Description
Criteria	Required	Select a field or formula in the dropdown list and enter values in the popup that appears. These filters are specific to workflow history. See the help topic Advanced Search Criteria Filters.
Results	Optional	Define display options for mass update results. The results appear when you preview the records for the mass update. See the help topic Search Results Display Options.
Audience	Optional	Define the users who can perform the mass update.
Schedule	Optional	Define a schedule on which to perform the update. See the help topic Scheduling a Mass Update.

The following screenshot shows the **Mass Updates** page for employee records created between two dates:



5. Click **Preview** to view the records with workflow instances to cancel:



6. Do one of the following:
   - To cancel the instances immediately, in the **Apply** column, select the records with workflow instances you want to cancel and click **Perform Update**. The status of the mass update is displayed on the **Mass Update Status** page.
   - To save the update and perform it later, click **Save**. NetSuite saves the mass update and displays the **Saved Mass Updates** page. You can return to this page to transition the records by going to Lists > Mass Update > Saved Mass Updates. Click **Preview** next to the name of the update. On the **Mass Update Preview** page, click **Perform Update** to cancel the instances.

Note: Optionally, from the **Mass Update** page, you can use the **Audit Trail** tab to view the changes to the current mass update or use the **Action Title Translation** tab to localize the title of the mass update.

Mass Transitioning Workflow Instances

You can transition an active workflow on a record outside of the defined workflow execution path. You can transition multiple instances of a specific workflow simultaneously for the same record type without requiring a predefined transition in the workflow definition. You should use the Mass Update feature to transition workflows in the following circumstances:

- The workflow ends up on the wrong state and you need a manual one-time fix.
- The workflow is out of sync with the record on which the workflow is running.
- The workflow gets stuck on a state in a workflow execution path.

Note: You should only mass transition workflow instances as a solution to a workflow problem and not as a way to regularly execute workflows.

To transition an active workflow on a record outside of the defined workflow execution path, you must be logged in as an Administrator or using a role with the correct permissions. The role needs to have Mass Updates permissions, Workflow Admin permissions, and permissions to edit the record type you are using.

To transition workflow instances:

1. Go to Lists > Mass Update > Mass Updates.
2. On the Mass Updates page, expand **Workflow**.

   For each record type in NetSuite, the Mass Updates page lists the workflows that are available to initiate, process, cancel, or transition.
3. Click **Transition <workflow name>** to transition the appropriate workflow.
4. If you are not going to perform the update immediately, on the **Mass Update** page, enter a title for the action in the **Title of Action** field.
5. In the **Transition To** dropdown list, select the state that you want to transition the workflow to.
6. If necessary, define the filter and other options on the following subtabs:

Subtab	Required / Optional	Description
Criteria	Optional	Select a field or formula in the dropdown list and enter values in the popup that appears. These filters are specific to workflow history.

Subtab	Required / Optional	Description
		See the help topic Advanced Search Criteria Filters.
Results	Optional	Define display options for mass update results. The results appear when you preview the records for the mass update. See the help topic Search Results Display Options.
Audience	Optional	Define the users who can perform the mass update.
Schedule	Optional	Define a schedule on which to perform the update. See the help topic Scheduling a Mass Update.

7. Do one of the following:
   - To perform the update immediately, in the **Apply** column, select the records you want to transition the workflow on and click **Perform Update**. The status of the mass update is displayed on the **Mass Update Status** page.
   - To save the update and execute it later, click **Save**. NetSuite saves the mass update and displays the **Saved Mass Updates** page. You can return to this page to transition the workflow instances by going to Lists > Mass Update > Saved Mass Updates. Click **Preview** next to the name of the update. On the Mass Update Preview page, click **Perform Update** to transition the workflow instances.

Note: Optionally, from the **Mass Update** page, you can use the **Audit Trail** tab to view the changes to the current mass update or use the **Action Title Translation** tab to localize the title of the mass update.

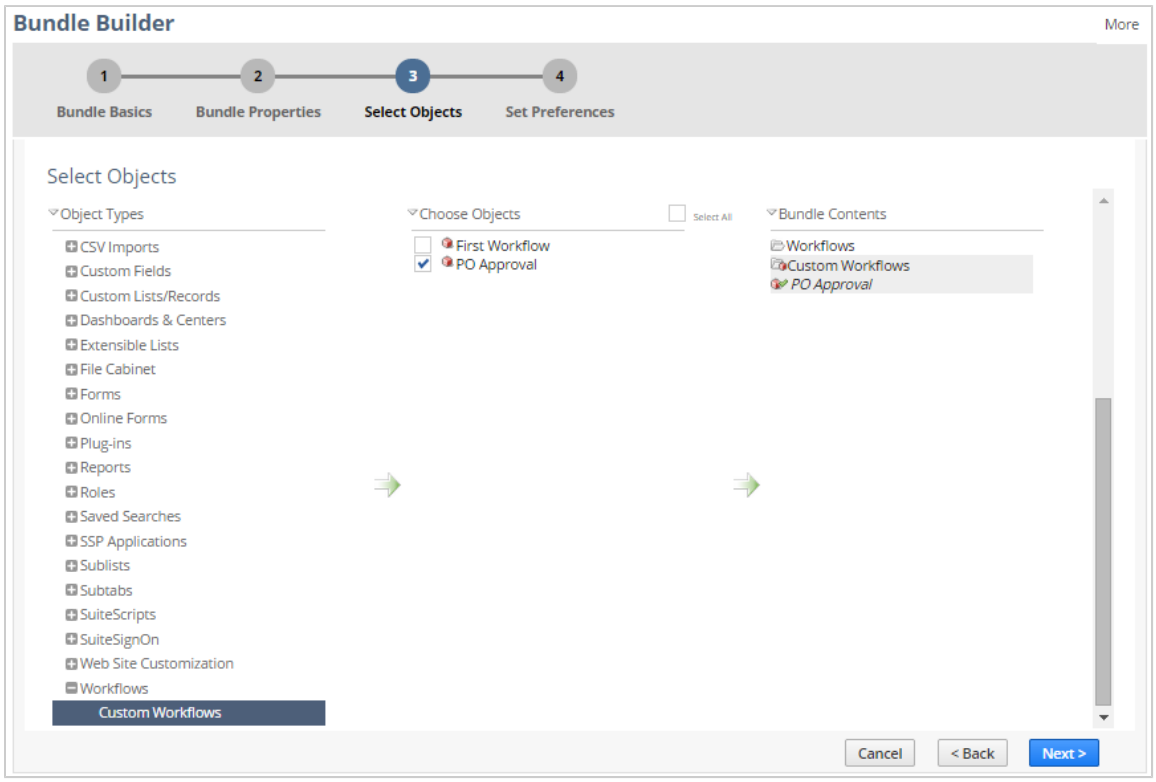
Bundling a Workflow

After you lay out and test a workflow, you can bundle the workflow to install the workflow into other NetSuite accounts. Workflow objects are lockable in a bundle. However, if you lock a bundled workflow, the workflow definition, custom fields, states, transitions, and actions cannot be edited in target accounts. Also, you cannot add states or transitions to a locked workflow, but you can add workflow and state fields.

Use the Bundle Builder to bundle a workflow. You can bundle any workflow in an account, including any inactive workflows. If you include a workflow that references a custom role, the custom role is automatically included. For more information about using the Bundle Builder, see the help topics SuiteBundler Overview and Creating a Bundle with the Bundle Builder.

Important: Workflow instances and history logs are not copied to target accounts when a bundle is installed or updated.

The following screenshot shows the Bundle Builder where you can select workflows to include in a bundle:



Rules and Guidelines for Bundling an Unlocked Workflow

Use the following rules and guidelines when working with bundled workflows:

- If states or transitions are modified in an unlocked workflow in the source account, NetSuite updates states and transitions previously installed in the target account with the states and transitions from the source account. NetSuite also adds any new states or transitions in the source account to the target account. If a state or transition with the same name exists in the source and target accounts, NetSuite duplicates the state or transition in the target account.

  For example, after you install a bundle, you create a state named **state10** in both source and target account and then update the bundle. In the target account workflow, two states named **state10** will appear, one created directly in the target account and the other installed by the bundle update.
- If states, transitions, or any other property of the workflow objects are modified in the target account, updating the bundle will revert the changes back to what the source dictates.
- If states or transitions are deleted in a target account, they will be re-added during a bundle update.
- If you delete any state or transition in the source, but use them in the target, you cannot remove these states or transitions in the target account if the workflow is running in the target account and is currently referencing that state. Bundle creators should advise their customers to make these states or transitions inactive if they are no longer intended to be used.

Editing a Locked Workflow

You can edit a workflow that has been locked after installation from a bundle using the Basic Information dialog. This dialog allows you to change the workflow release status and to enable or disable logging for the workflow.

To edit a locked workflow:

1. Go to Customization > Workflow > Workflows.
2. On the Workflows page, click **Edit** next to a workflow that is locked. The Basic Information dialog appears.
3. To edit the release status of the dialog, select the appropriate status from the **Release Status** dropdown list.
4. To enable or disable logging for the workflow, check or clear the **Enable Logging** box. When logging is enabled for the locked workflow, you can view log information about the workflow record's System Information > Workflow History subtab.
5. To specify when to keep workflow history, in the **Keep Instance and History** dropdown list, click **Always**, **Never**, or **Only When Testing**. For details, see Disabling History for a Workflow.

Note: A padlock icon indicates that a workflow is locked on the Workflows page.

SuiteFlow Reference and Examples

Use this reference information and examples to further develop and customize a workflow. Use this information when you create or edit workflow definitions, states, actions, and transitions.

Note: For more information about creating and editing workflow definitions, see Working with Workflows.

Workflow Elements Reference

The following table lists the reference information for the elements of a workflow. Use the reference information to further develop and customize a workflow.

Workflow Element	Reference Topics
Actions	Workflow Actions Overview
Triggers	Triggers Reference
States	States Reference
Event types	Event Types Reference
Contexts	Execution Context Types
Workflow Templates	Workflow Templates Reference

Workflow Elements and Features Examples

The following table lists the examples for using the different elements of a workflow and SuiteFlow features. Use the examples to further develop and customize a workflow.

Workflow Element	Example Topics	Included Examples
Actions	Action Examples	■ Using Buttons to Execute Transitions ■ Using Buttons for Navigation ■ Executing an Action with a Saved Search Condition ■ Using Conditional Fields with Actions ■ Setting Field Values in Action Definitions ■ Creating and Subscribing to a Record
Transitions	Transitions Examples	■ Blank Transition Trigger ■ Executing a Transition with a Saved Search Condition ■ Specifying States for Child Workflow Transitions
Conditions	Condition Examples	■ Referencing Old (Pre-edit) Values in a Workflow ■ Defining Conditions for Customer Credit Hold Field
Context types	Context Type Examples	■ CSV Import ■ User Event Script ■ Custom Mass Update
Event types	Examples of Event Types	■ Print Event Type

Workflow Element	Example Topics	Included Examples
		■ Mark Complete Event Type ■ Drop Ship Event Type
Search	Workflow Search Examples	■ Using the Button Filter in a Workflow Instance Search

Workflow Actions Overview

Actions are specific tasks performed by a workflow instance based on the properties of a record. You can use actions in workflows to manipulate the fields on a record, add or remove buttons, create new records, send email, and more. After you create a state in a workflow, you choose the actions that you want to execute when the record is in that state. When a record enters a state in a workflow, the workflow instance executes the actions depending on their triggers.

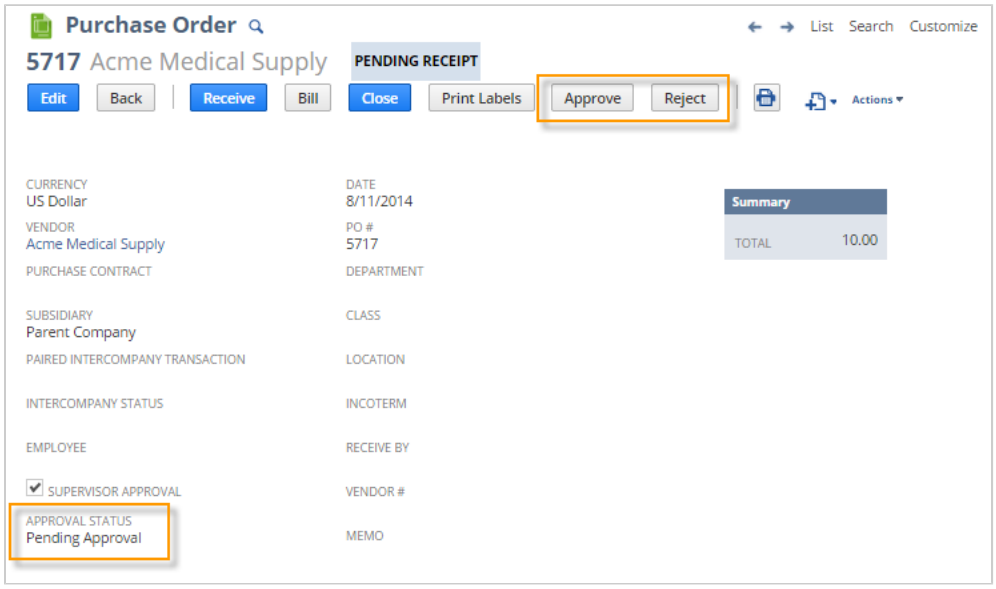
Action	Description	More Information
Add Button	Add a button to a record form.	Add Button Action.
Confirm	Display a popup message with OK and Cancel buttons.	Confirm Action.
Create Line	Create a new line in a sublist.	Create Line Action.
Create Record	Create a new record.	Create Record Action.
Go To Page	Redirect users from a record in a workflow to a specified page in NetSuite.	Go To Page Action.
Go To Record	Redirect users to a specific record in NetSuite.	Go To Record Action.
Initiate Workflow	Initiate another workflow instance from the current workflow instance.	Initiate Workflow Action.
Lock Record	Lock the record for the workflow instance.	Lock Record Action.
Remove Button	Remove a button from a record form.	Remove Button Action.
Return User Error	Display an error message to users.	Return User Error Action.
Send Campaign Email	Send an email as part of a marketing initiative.	Send Campaign Email Action.
Send Email	Send an email.	Send Email Action.
Set Field Display Label	Change the UI label of a field in a workflow or sublist.	Set Field Display Label Action.
Set Field Display Type	Change the display type of a field.	Set Field Display Type Action.
Set Field Mandatory	Require a field in a workflow or sublist to have a value.	Set Field Mandatory Action.
Set Field Value	Set the value of a record field.	Set Field Value Action.
Show Message	Display a message to the user.	Show Message Action.
Subscribe To Record	Track a workflow or state field that contains a reference to a record.	Subscribe To Record Action.
Transform Record	Transform the data on a transaction record into another transaction record type.	Transform Record Action.
Custom	Define an action with a Workflow action script.	Custom Action.

Add Button Action

The Add Button action adds a button to a record form. You can use the Add Button action to create a button to do any of the following:

- Transition a record to the next state in a workflow. For more information, see Using Buttons to Execute Transitions.
- Go to another page or record. For more information, see Using Buttons for Navigation.
- Initiate mass processing on workflow instances for a specific base record type. See Mass Processing Records in a Workflow.

The following screenshot shows an example of using the Add Button action to create buttons that transition the record to the next state in a workflow. In the following example, the Purchase Order record includes Approve and Reject buttons. Users can use the Approve and Reject buttons to transition to the next state in the workflow, and then change the Approval Status field of the purchase order.



Add Button Action Triggers

The Add Button action supports the following server-side triggers:

- Before Record Load (default trigger)
- On Entry
- On Exit

For more information about triggers and the Add Button action, see Workflow Triggers Quick Reference.

Add Button Action Parameters

Each workflow action contains parameters specific to that action that you need to specify when you create the action. The following table describes the parameters specific to the Add Button action:

Parameter	Description	Additional Information
Label	Text that represents the button on the record form.	This parameter is required.

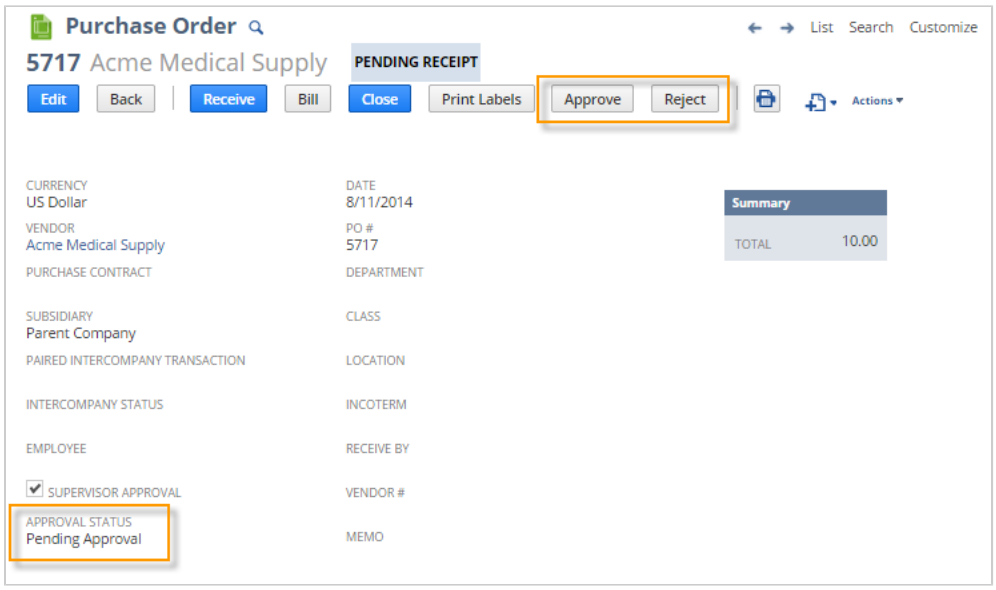
Parameter	Description	Additional Information
Save Record First	Select this option to save the record before the action is executed.	This parameter is disabled by default. When you enable this option and then edit the record, the button label is added as an option to the Save button on the record form, using the format Save & *<button label>*. For example, if you add a button with the label Accept and you check the Save Record First box, the Save button is amended to Save & Accept.
Check Condition Before Execution	Select this option to evaluate the condition associated with an Add Button action before the action associated with the button executes.	This parameter is enabled by default. When you enable this option, the workflow evaluates the conditions and verifies that the record exists in the saved search results before executing the Add Button action. If you check the Save Record First box, the workflow uses any changed property for the record after a record is saved when it evaluates the conditions. If the conditions are evaluated as false by the workflow or if the workflow cannot evaluate the conditions, the workflow does not execute the action.

Add Button Action Guidelines

Use the Add Button action to add a button to a record form. Use the Add Button action to create a button that transitions the record to the next state in a workflow or go to another page or record.

If the Multi-Language feature is enabled on your account, you can enter translations for the button label. For more information, see Entering translated strings for button labels.

The following screenshot shows a Purchase Order record with **Approve** and **Reject** buttons added with the Add Button action. Users can use the **Approve** or **Reject** buttons to transition to the next state in the workflow and then change the **Approval Status** field of the purchase order.



See the following examples for more information about using the Add Button action:

- Using Buttons to Execute Transitions
- Using Buttons for Navigation

You can also use a button to initiate mass processing on workflow instances for a specific base record type. See Mass Processing Records in a Workflow.

Add Button Action Properties

The following screenshot shows the **Parameters** section for the Add Button action:

The following table describes the Add Button action properties:

Property	Description
Label	Text that appears for the button on the record form. You must set this option.
Save record first	Indicates whether the record should always be saved before the action is executed. This option is disabled by default. When you enable this option and then edit the record, the button for the action is added as a sub-option of the Save button on the record form, with a label of **Save &** *<Action Label>*. NetSuite saves the record before the action associated with the button executes. If you view the record, the button always appears as a separate button on the record form.
Check condition before execution	Indicates whether the workflow evaluates the condition associated with an Add Button action before the action associated with the button executes. This option is enabled by default. When this option is enabled, the workflow evaluates the conditions and verifies that the record exists in the saved search results before executing the Add Button action. If the **Save record first option** is enabled, the workflow uses any changed property for the record after a record is saved, when it evaluates the conditions. If the conditions evaluate to false, or if it cannot be evaluated, the workflow does not execute the action. Use the following guidelines: ■ The **Check condition before execution** property has no effect on an Add Button action that does not have a condition or saved search defined. In this case, the check evaluates to true and the workflow executes the action. ■ In cases where the **Save record first** and **Check condition before execution** properties are both enabled, when the **Save &** *<Action Label>* button is clicked, the workflow saves the record and then the workflow evaluates the condition.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Add Button Action Guidelines

Important: Use the following guidelines when working with the Add Button action.

- If you use the Add Button action to add buttons to the record form in a workflow and you want the buttons to appear every time a user accesses the record, make sure that the Add Button actions trigger on the Before Record Load trigger, and not on the Entry trigger.
- If a button should appear in multiple states, add the Add Button action to each state. For example, if you add the Add Button action for a **Close Order** button in State 1, add or copy the **Close Order** button action to State 2, and all other states where the button should appear. The button appears in each state when a user loads the record.

- You cannot use the Remove Button action to remove buttons added with the Add Button action. To remove the button, delete the Add Button action or set it to inactive.
- If you use the Add Button action to add buttons to a newly created record (triggered On Create), the buttons are disabled on the new record form. The newly added buttons display, but are grayed out and cannot be clicked by users.
- If you create a workflow that includes an Add Button action, when a user interacts with the button and the record is saved, the System Notes will list the role of the user as Administrator, regardless of the role the user is logged in as. This behavior is expected and is not related to the Execute As Admin property on the Workflow Definition page.

Entering translated strings for button labels

Every company has a default language that is set by an Administrator or a user with the Set Up Company permission. The default company language can only be changed by an Administrator or a user with the Set Up Company permission. However, if the Multi-Language feature is enabled in your account, you can enter translations for button labels on the Workflow Action page regardless of the default company language.

You can add languages to your account on the General Preferences page. Any languages that you add to your account will be added to the translation table on the Workflow Action page. For more information, see the help topic Configuring Multiple Languages.

1. On the **Workflow Action** page, enter the translated label in the **Name** column, next to the appropriate language.
2. If you do not want to enter a translation for a language, leave the **Name** column next to the language empty. If you do not enter a translated label, the default company language will be used for the button label.
3. When you are done entering translated labels, click **Save**.

About Multiple Clicked Workflow Buttons

SuiteFlow prevents workflow instances and actions from processing simultaneously in the following scenarios:

- A user double-clicks a workflow button.
- A user clicks workflow button A, then clicks workflow button B before the record page automatically refreshes.
- A user clicks workflow button A, then clicks workflow button B, then clicks workflow button X before the record page automatically refreshes.

Note: A workflow button is a button added by an Add Button action.

When users double-click a workflow button, or click a sequence of workflow buttons, SuiteFlow alerts the user and asks if they would like to start processing the action when the previous action is still in progress.

Confirm Action

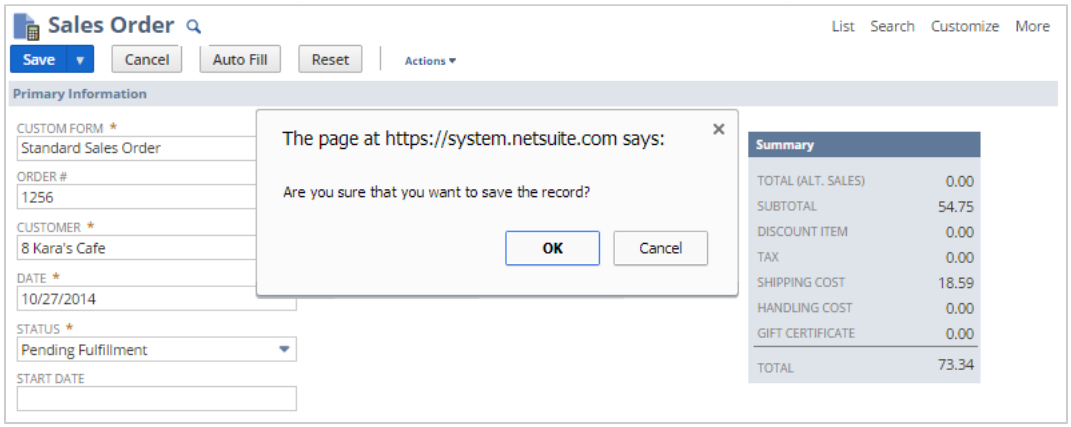
Use the Confirm action to display a popup message with **OK** and **Cancel** buttons. A user can acknowledge the message and continue working or cancel the previous activity. For more information about adding actions to a workflow, see Creating an Action.

If the Multi-Language feature is enabled on your account, you can enter translations for confirmation messages. For more information, see Entering translated strings for confirmation messages.

The Confirm action supports the Before Field Edit and Before User Submit client triggers. The behavior of the buttons for the Confirm action depends on the type of trigger used:

Client Trigger	Button Behavior
Before Field Edit	■ **OK.** The field value remains set. ■ **Cancel.** The field value reverts to the previous value.
Before User Submit	■ **OK.** NetSuite saves the record. ■ **Cancel.** NetSuite does not save the record. The user can continue editing the record.

The following screenshot shows the Confirm action on a Before User Submit trigger:



The Confirm action is a combination of the functionality of the Show Message action and the Return User Error action. Both Show Message and Confirm use the **OK** button to allow a user to acknowledge the message. Both Return User Error and Confirm use the **Cancel** button to prevent the user from moving forward. See Return User Error Action and Show Message Action.

Confirm Action Properties

The following screenshot shows the **Parameters** section for the Confirm action:

The following table describes the Confirm action properties:

Property	Description
Text	Text to appear in the popup window when the Confirm action executes.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Entering translated strings for confirmation messages

Every company has a default language that is set by an Administrator or a user with the Set Up Company permission. The default company language can only be changed by an Administrator or a user with the

Set Up Company permission. However, if the Multi-Language feature is enabled in your account, you can enter translations for confirmation messages on the Workflow Action page regardless of the default company language.

You can add languages to your account on the General Preferences page. Any languages that you add to your account will be added to the translation table on the Workflow Action page. For more information, see the help topic Configuring Multiple Languages.

1. On the **Workflow Action** page, enter the translated label in the **Translation** column, next to the appropriate language.
2. If you do not want to enter a translation for a language, leave the **Translation** column next to the language empty. If you do not enter a translated label, the default company language will be used for the button label.
3. When you are done entering translated confirmation messages, click **Save**.

Create Line Action

Create Line Item in SuiteFlow

Use the Create Line action to add a new line to a sublist in a workflow. The Create Line action enables you to customize sublist-related processes by adding lines to a sublist as part of your workflow. You can specify whether the new line is added as the first or last line in the sublist. You can also use the Create Line action to specify default values for line fields, and the values can be static or dynamically defined. The Create Line action can be added to a workflow state as an individual action or as part of an action group.

The Create Line action is available in workflow definitions on record types with a sublist SuiteFlow supports. Currently, SuiteFlow only supports the Items sublist on Transaction record types.

The Create Line action only supports server side triggers. The following table describes the valid triggers for the Create Line action:

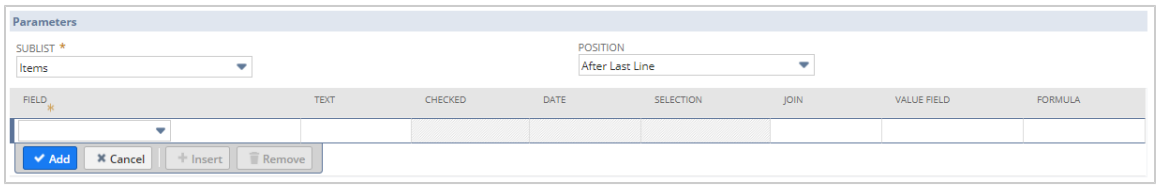
Trigger	Description
Entry	Action executes the first time a record enters a state in a workflow.
Exit	Action executes the first time a record exits a state in a workflow.
Before Record Load	Action executes before the record form loads into the browser.
Before Record Submit	Action executes after a user clicks **Save** on a record but before NetSuite saves the record to the database.
After Record Submit	Action executes after NetSuite saves the record to the database.
Scheduled	Action executes on a schedule. For more information, see Scheduling an Action.

Note: The Create Line action does not support client triggers.

Create Line Action Properties

When you create a Create Line action, you specify the type of line you are creating and the position in the sublist where you want to create the line. The parameters you need to complete will vary depending on the type of line you are creating.

The following screenshot shows the Create Line action parameters:



The following table describes the Create Line action parameters:

Parameter	Description
Sublist	The type of sublist that you want to create the line in. Currently, SuiteFlow only supports the Items sublist on Transaction Record types.
Position	The position in the sublist where you want to create a line. Options include **After Last Line** and **Before First Line**.
Field	The type of line that you want to create. For each type of line that you create, complete the parameters as necessary, and click **Add**.
Text	The text of the message to display to the user.
Checked	Select this option to include a checkbox with the line.
Date	Select this option to include a date field.
Selection	Select this option to include a selection field.
Join	Select the ID of a record to redirect to with a join.
Value Field	Enter the default value for the line field.
Formula	Enter the default formula for the line field.

Create Record Action

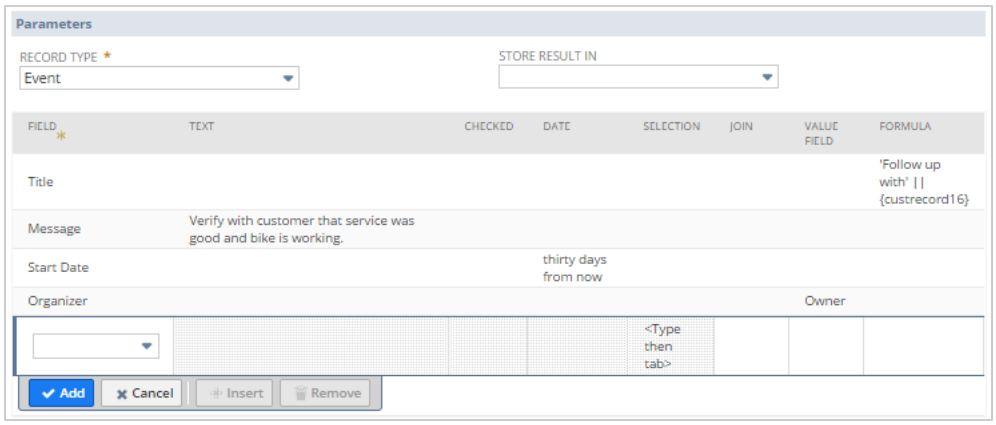
Use the Create Record action to create a new record in NetSuite. The record can be standard NetSuite record or a custom record. The record does not need to be the same record type as the base record type for the workflow.

Use this action, for example, to create a record that must be followed up on as part of the business process for the workflow. Or, you can use the Create Record action to trigger the execution of another workflow, if the other workflow initiates on the record created by the Create Record action. The Create Record can also be used with the Subscribe To Record action. Create a record and then create conditions based on the state of the created record. See Creating and Subscribing to a Record.

Create Record Action Properties

When you create a Create Record action, you specify the type of record to create and then specify the field values to populate on the record to be created. These properties differ depending on the type of record selected in the **Record Type** dropdown list.

The following screenshot shows the Create Record action properties when creating an Event record:



Note: In the above example, the **{custrecord16}** variable and the **Owner** value for the **Organizer** field correspond to values on the record in the workflow, not on the record to create.

The following table describes the Create Record action properties:

Property	Description
Record Type	Type of record to create. The list of properties in the **Field** column changes depending on the type of record.
Store Result In	Workflow field or state field in which to store a reference to the created record. This field must exist before you create the action. Optional. Use this field to subscribe to a created record with the Subscribe To Record action. See Creating and Subscribing to a Record.
Field	Field values to populate when creating the record in the **Record Type** dropdown list. For each field, select the field name, set the appropriate properties in the other columns, and click **Add**. For more information, see Setting Field Values in Action Definitions **Note:** These are not field values for the base record type of the workflow.

Create Record Action Guidelines

Important: Use the following guidelines when working with the Create Record action.

- When using the Create Record action, you must set all required fields for the record type. To verify the required fields, you can create a new record in NetSuite and note which fields are required.

  If you do not set all required fields, the Create Record action fails.
- The Create Record action is not available on a Before Record Load trigger for a time entry record.

Custom Action

Use the Custom action to define an action with a Workflow Action script. The functionality for a custom action is defined with SuiteScript in a Workflow Action script and created like other action types.

For information about SuiteFlow workflow action scripts and to see sample code, see the help topic SuiteScript 2.x Workflow Action Script Type.

To create a custom action, complete the following basic steps:

1. Create and deploy a Workflow Action script to define the functionality of the action.
2. In the Workflow Manager, create a new action. Deployed Workflow Action scripts appear on the **New Action** window. The custom action only appears if the script was deployed for the base record type of the workflow. See New Action Window for a Custom Action.

   Each custom action includes the common **Basic Properties**, **Condition**, and **Schedule** sections.

   See Creating an Action.
3. Set the custom action parameters. Depending on the functionality defined in SuiteScript, you can choose to store the result of the Workflow Action script in a custom field when you create the action in the Workflow Manager. See Custom Action Properties.

Note: For an example of a custom action, see Storing a Return Value from a Custom Action Script in a Workflow Field.

Custom Action Guidelines

Important: Use the following guidelines when working with custom actions.

- You should test all custom action scripts independent of the workflow. If the script runs successfully, move the script deployment status from Testing to Released when you release the workflow.

  If the workflow action script deployment is not set to Released when the scripts runs in a workflow, the script will only run for only the script owner. The deployment status for both the custom action script and the workflow should match.
- For a custom action to be available to all workflows, set the **Applies To** property on the **Deployments** tab of the Workflow Action script record to **All Records**.
- Custom actions are skipped in workflows initiated by users outside of the custom script's specified audience. The skipped action scripts are listed in the Workflow Execution Log.

Workflow Nesting Guidelines

Workflow nesting occurs when you add a custom action to your workflow that invokes a record-changing event on records that are being processed by the workflow. Examples of record-changing actions include the following:

- Actions that load a record
- Actions that set a field value
- Actions that submit a record

When you add a custom action that invokes a record-changing event, it can cause a chain of workflow executions and customized scripts where the same workflow is executed again before the first execution is complete. This nesting renders the workflow vulnerable to errors and consequently the workflow may not execute properly.

Important: For optimal performance, you should avoid designing workflows with nested workflow executions. Workflows that are designed to rely on nested execution can negatively impact NetSuite performance.

Custom Actions and Workflow Governance

You create custom actions in workflows by using a Workflow Action script. When you create a custom action, the custom action should not exceed the following limits:

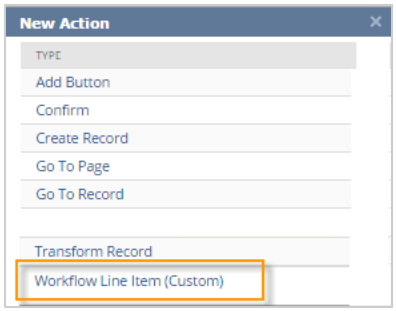
- Time limit: 300 seconds
- Line limit: 100,000,000
- Unit limit: 1,000

Note: The limits listed above apply to each custom action and are used by SuiteScript to implement the custom action. If your workflow contains multiple custom actions, the limits apply to each instance of a custom action and not the total of all of the custom actions in the workflow.

For more information about line limitations in NetSuite, see the help topic Line Limitations for Transactions.

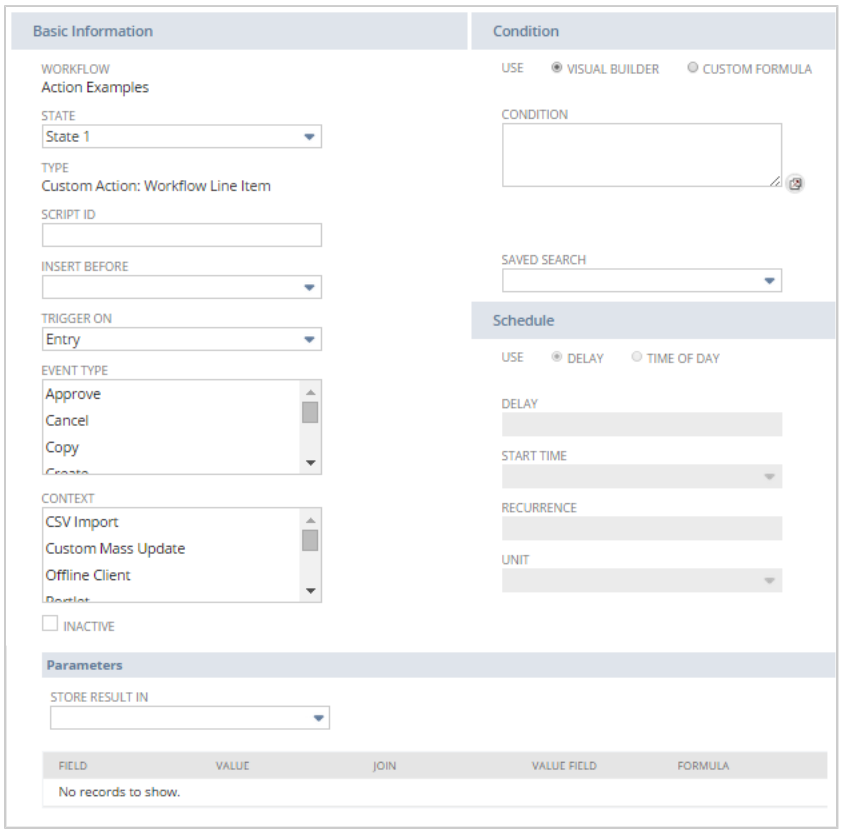
New Action Window for a Custom Action

The following screenshot shows an excerpt from the **New Action** window for a Workflow Action script named **Workflow Line Item**, deployed on a Sales Order record:



Custom Action Properties

The following screenshot shows the common properties for all custom actions:



Note: For more information about these properties, see Creating an Action.

Go To Page Action

Use the Go To Page action to redirect users from a record in a workflow to a specified page in NetSuite when the action executes. Pages in NetSuite to which you can redirect include list views or reports. To use the Go To Page action, select the page or report to redirect users.

Go To Page Action Properties

The following screenshot shows the **Parameters** section for the Go To Page action:

The following table describes the Go To Page action properties:

Property	Description
Page	NetSuite page or report to redirect to when the action executes. Type the name and press the Tab key to open a popup window that contains only records that begin with the letters you typed, or click the icon to open a popup list to select the page or report.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Go To Page Action Guidelines

Important: Use the following guidelines when working with the Go To Page action.

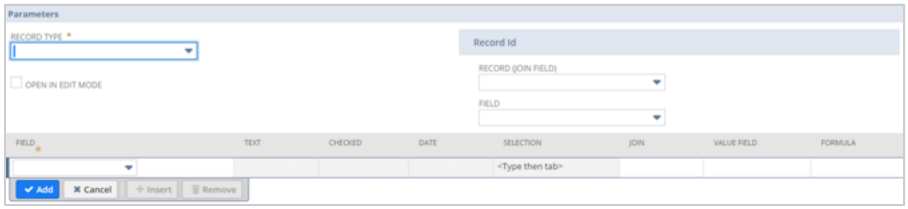
- If you want to redirect to a specific record instead of a list page or report, use the Go To Record action. See Go To Record Action.
- The Go To Page action is not available on a Before Record Submit or After Record Submit trigger for a time entry record.

Go To Record Action

Use the Go To Record action to redirect users to a specific record in NetSuite. To use the Go To Record action, set the record type to redirect to when the action executes and an optional ID field that uniquely identifies the record. If you do not specify a record ID, then NetSuite creates a record with the properties you select.

Go To Record Action Properties

The following screenshot shows the **Parameters** section for the Go To Record action:



The following table describes the Go To Record action properties and the values you must use to create a record or redirect to a specific record:

Property	Create | Redirect	Description
Record Type	Create a new record Redirect to a record	Type of record to redirect to or create. The list of properties in the **Field** column changes depending on the type of record.
Open in edit mode	Create a new record	Indicates that the record to which the user is redirected opens in edit mode. This property only applies if you specify a value in the ID field. Optional.
Record (Join Field)	Redirect to a record	Use this field to get the ID of a record to redirect to with a join. Optional.
Field	Create a new record Redirect to a record	Field values to populate when creating the record in the **Record Type** dropdown list. For each field, select the

Property	Create | Redirect	Description
	Note: Also see Go To Record Action Guidelines.	field name, set the appropriate properties in the other columns, and click **Add**. For more information, see Setting Field Values in Action Definitions.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Go To Record Action Guidelines

Important: Use the following guidelines when working with the Go To Record action.

- You cannot set field values on an existing record if the Go To Record action executes on a Before Record Load trigger. This limitation applies to Set Field Value and Go To Record actions. Instead, use a custom action to update fields on an existing record, then use the Go To Record action to go to the record after it has been updated.
- If you do not use the field definition area to set field values, on any SuiteScript supported record you can add a URL parameter such as `record.{field}={value}` and the system will initialize that field, as if you had called `setFieldValue` in a before load event handler.
- The Go To Record action is not available on a Before Record Submit or After Record Submit trigger for a time entry record.

Initiate Workflow Action

Use the Initiate Workflow action to initiate another workflow instance from the current workflow instance to process multiple workflows in parallel. The workflow instance initiated with the Initiate Workflow action, or child workflow, runs on the same record instance as the parent workflow instance. For example, if a workflow instance running on a purchase order initiates a child workflow instance with the Initiate Workflow action, both workflow instances run on the same purchase order.

You can populate workflow fields on the child workflow with values from the current workflow record or values available through a join.

You can also use the completion of the child workflow as a condition for a transition to execute. Select the child workflow definition on the **Workflow** field for a transition. When the child workflow completes, the transition executes. For more information about the best practices to follow when using the Initiate Workflow action, see SuiteFlow Best Practices. See Specifying States for Child Workflow Transitions.

Initiate Workflow Action Properties

The following screenshot shows the **Parameters** section for the Initiate Workflow action. In this example, the action populates the workflow field Rate on the child workflow with the Tax Rate of the current record in the workflow.

The following table describes the Initiate Workflow action properties and the columns used to specify workflow field values:

Property	Description
Workflow	Name of the workflow definition to initiate when the action executes. This list shows all workflows with the same base record type as the current workflow definition that are not inactive.
Field	Workflow field values to populate when initiating a child instance of the workflow in the **Workflow** dropdown list. Each workflow field defined for the child workflow appears in this column. For each field, set the appropriate properties in the other columns. For more information about each column, see Setting Field Values in Action Definitions.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Initiate Workflow Action Guidelines

Important: Use the following guidelines when working with the Initiate Workflow action.

The Initiate Workflow action is not available on a Before Record Load trigger for a time entry record.

Lock Record Action

Use the Lock Record action to lock the record for the workflow instance when the record is in the state. After the action executes, the **Edit** button is removed from the record in the browser. To lock a record when the record is in multiple states, add the action to each state.

The Lock Record does not have any additional parameters. For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Lock Record Action Guidelines

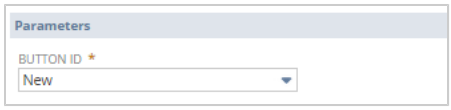
Important: Use the following guidelines when working with the Lock Record action.

- If you attempt to load a locked record in SOAP web services or SuiteScript, NetSuite throws an error and does not load the record.
- The default trigger type for the Lock Record action is the Before Record Load trigger. Although the Entry and Exit triggers are available, best practice is to lock the record before it loads into the browser.
- Locked records still appear with **Edit** in a list, but cannot be edited after they are loaded into the browser.
- The Lock Record action is not available on a Before Record Load trigger for a time entry record.

Remove Button Action

Use the Remove Button action to remove a button from a record form in a workflow instance. Use this action to remove both standard NetSuite buttons and custom buttons, based on conditions such as the user that views the record or the state of the record.

The following screenshot shows the **Parameters** section for the Remove Button action. In this example, the action removes the **New** button from the current record in the workflow.



The following table describes the Remove Button action properties:

Property	Description
Button ID	ID of the button to remove on the record form. The button ID can be from a standard NetSuite button or a custom button.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Remove Button Action Guidelines

Important: Use the following guidelines when working with the Remove Button action.

- You cannot use the Remove Button action to remove a button added with the Add Button action. To remove the button, delete or inactivate the Add Button action.
- The default trigger type for the Remove Button action is the Before Record Load trigger. Although the Entry and Exit triggers are available, best practice is to remove the button before it loads into the browser. In the Workflow FAQ section, see Q: .

Return User Error Action

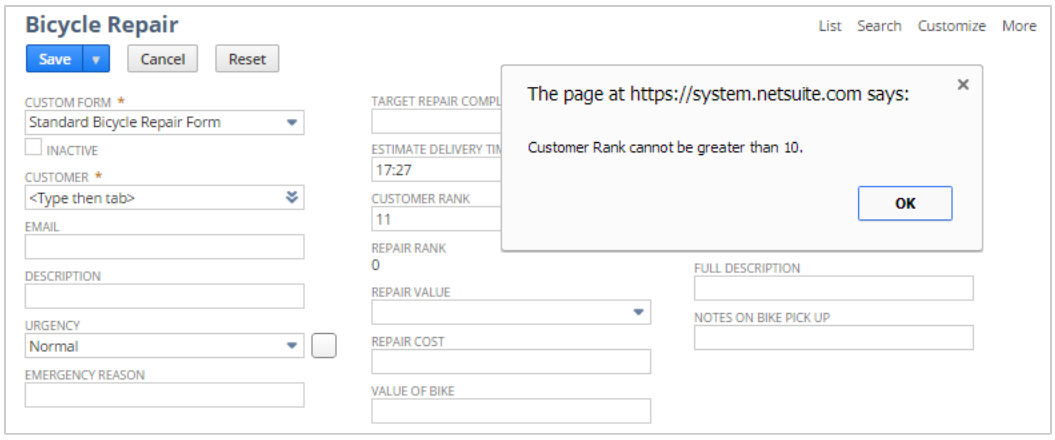
Use the Return User Error action to display an error message to users. Use this action with field or record validation to display incorrect data on the record form to the user, and prevent the user from saving the record with incorrect or invalid data. To create the Return User Error action, create a condition on the action definition to identify the error and then set the message text in the **Text** box.

If the multi-language feature is enabled on your account, you can enter translations for error messages. For more information, see Entering translated strings for error messages.

The error message displayed to the user and the usage of the action depends on the type of trigger used with the Return User Error action:

Trigger Type	Use Case	Error Message Format
Before Record Load	Validate record form field values before the record loads into the browser.	Displays as browser page with a **Go Back** button.
Any client trigger	Validate record form field values as user interacts with the record form.	Displays as popup window with an **OK** button.
Before Record Submit	Validate record form field values before the record is saved to the database, after the user clicks **Save**.	Displays as browser page with a **Go Back** button.

The following screenshot shows the execution of the Return User Error action on a client trigger:



The following screenshot shows the execution of the Return User Error action on a server trigger:

Important: When the user clicks **Go Back**, all field values entered since the record was last saved are removed. The user must re-enter all data. This prevents the user from saving the record until all data is correct and valid. See Return User Error Action Guidelines.

Return User Error Action Properties

The following table describes the Return User Error action properties:

Property	Description
Text	Text of the message to display to the user.

Note: For more information about adding actions to a workflow, including common action properties and conditions, see Action Conditions and Creating an Action.

Return User Error Action Guidelines

Important: Use the following guidelines when working with the Return User Error action.

- From a user experience perspective, it is more efficient and less frustrating to detect errors on individual fields as they are entered, rather than after the user saves the record. If you execute the action on a server trigger, the user must re-enter all data since the last successful save.
- The Return User Error action is not available on a Before Record Load trigger for a time entry record.

Entering translated strings for error messages

Every company has a default language that is set by an Administrator or a user with the Set Up Company permission. The default company language can only be changed by an Administrator or a user with the Set Up Company permission. However, if the Multi-Language feature is enabled in your account, you can enter translations for button labels on the Workflow Action page regardless of the default company language.

You can add languages to your account on the General Preferences page. Any languages that you add to your account will be added to the translation table on the Workflow Action page. For more information, see the help topic Configuring Multiple Languages.

1. On the **Workflow Action** page, enter the translated label in the **Translation** column, next to the appropriate language.
2. If you do not want to enter a translation for a language, leave the **Translation** column next to the language empty. If you do not enter a translated label, the default company language will be used for the button label.
3. When you are done entering translated error messages, click **Save**.

Send Campaign Email Action

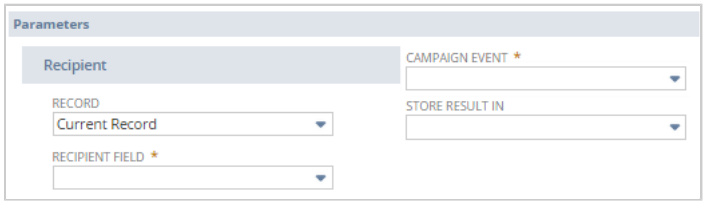
Use the Send Campaign Email action to send an email as part of a marketing initiative. This action can be used in a workflow for any record type, although it is commonly used with Customer, Lead, or Prospect record types. You can use the Send Email Campaign action in a lead nurturing workflow to manage leads in a marketing campaign. For a sample workflow that uses a Send Campaign Email action, see Lead Nurturing Workflow.

Use this action in conjunction with the Marketing Automation feature. Enable the feature on the CRM subtab at Setup > Company > Enable Features.

To use the Send Campaign Email action, you specify the recipient of the email and the campaign event, and an optional workflow field to track the customer responses to the campaign event. Use the Subscribe To Record action to track responses to the campaign event. For more information about campaign events, see the help topic Managing Campaigns.

Send Campaign Email Action Properties

The following screenshot shows the Send Campaign Email Action properties:



The following table describes the Send Campaign Email action properties:

Property	Description
Record	Record that contains the recipient to use with the campaign event. Default is **Current Record**.
Recipient Field	Recipient of the campaign email. NetSuite uses the email address of the entity in this field. For example, for **Customer**, NetSuite emails the campaign event to the customer on the current record.

Property	Description
Campaign Event	Title of the campaign event for the associated marketing campaign. Create a campaign event from the **Events** subtab on a Marketing Campaign record. See the help topic Campaign Events.
Store Result In	Workflow field to store the campaign event response. Use this field in a Subscribe To Record action to track the response from the recipient of the campaign email. See Creating and Using Workflow Fields and Subscribe To Record Action.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Send Email Action

Use the Send Email action to send an email when the action executes. When you create a Send Email action, you specify the sender and the recipient, the content of the email as custom text or as a template, and any additional information to attach to the email. If the recipient is also a NetSuite customer, you can include a link to the current record in the workflow.

You can also schedule a Send Email action. You can use the **Scheduled** trigger and then specify a delay for the email to be sent on the action definition. You can also schedule a Send Email action by scheduling a transition into a state that contains a Send Email action. See Scheduling an Action and Scheduling a Transition.

The following table describes the additional features of the Send Email action that you can use to customize the email:

Feature	Description
Use internal NetSuite IDs	Use NetSuite internal IDs to dynamically generate field values. See Using Internal IDs.
Include attachments	Add NetSuite-supported file types as an attachment. See Attaching Files to an Email.
Include statements	Include statements for workflows based on the Customer, Lead, and Prospect record types. See Attaching Statements to an Email.
Include transactions	For workflows based on a Transaction record type, include a record in the body of the email or as an attachment. See Attaching Transactions to an Email.

Send Email Action Triggers

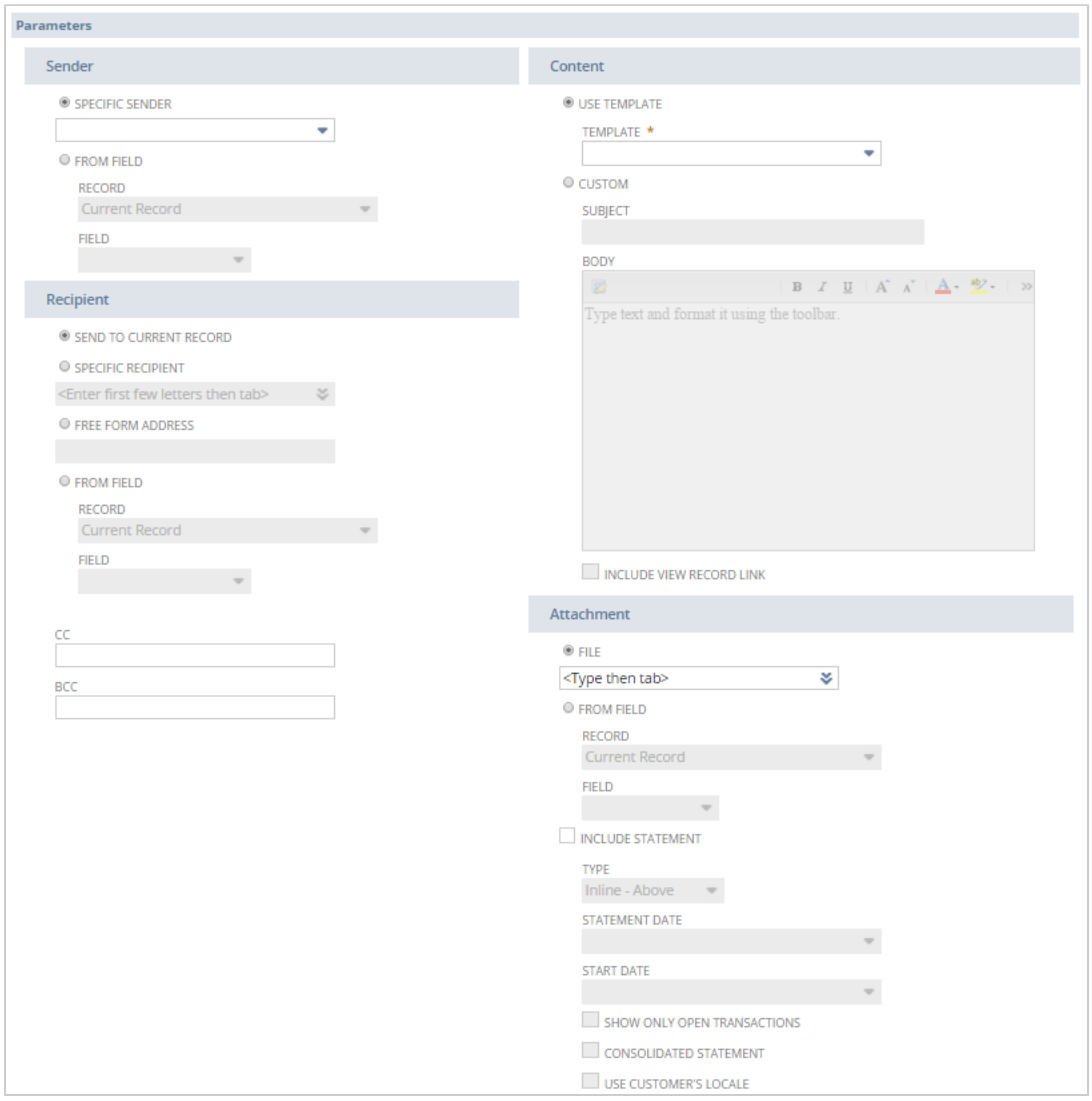
The triggers used with the Send Email action determines when the email is sent. The following table describes the valid triggers for a Send Email action:

Trigger	Description
Entry	Executes the first time a record enters a state.
Exit	Executes the first time a record exits a state in a workflow.
After Record Submit	Executes after NetSuite saves the record to the database.
Scheduled	Executes on a schedule. See Scheduling an Action.

Important: If a Send Email action uses an Entry, it may not execute, depending on the trigger on which the record entered the state in the workflow. For more information about testing a Send Email action, see Testing Actions and Transitions and SuiteFlow Trigger Execution Model.

Send Email Action Properties

The following screenshot shows **Parameters** section of the Send Email action properties for a workflow based on a Customer record. The options available in the **Attachment** section differ depending on the type of record.



The following table describes the Send Email action properties by section type:

Section	Property	Description
Sender	Specific Sender	Use the email address associated with a specific NetSuite account and the sender.
	From Field	Set the following properties:

Section	Property	Description
		▪ **Record.** Select **Current Record** or another joined record type that contains the field with the associated email address to use as the sender. ▪ **Field.** Use the email address associated with this field. It can be a field that references an email address or a field that references another record.
Recipient	Send To Current Record	Send to the email address associated with the current record in the workflow.
	Specific Recipient	Use the email address associated with a specific NetSuite account as the recipient.
	Free Form Address	Email address or addresses to use. Separate multiple address with a comma, with no spaces in between.
	From Field	Set the following properties: ▪ **Record.** Select **Current Record** or another joined record type that contains the field with the associated email address to use as the recipient. ▪ **Field.** Use the email address associated with this field. It can be a field that references an email address or a field that references another record.
	Cc	Email address or addresses to use as a **Cc**. Separate multiple address with a comma, with no spaces in between. Field is limited to 1000 characters. You can also reference internal NetSuite IDs. See Using Internal IDs.
	Bcc	Email address or addresses to use as a **Bcc**. Separate multiple address with a comma, with no spaces in between. Field is limited to 1000 characters. You can also reference internal NetSuite IDs. See Using Internal IDs.
Content	Use Template	Select an email template to use for the email content. You must have the Mail Merge feature enabled to use email templates. **Important:** You can use CRMSDK tags in the email template, but not internal NetSuite IDs. See the help topics Using CRMSDK Tags and Working with Email Templates.
	Custom	Select this option and manually enter the content in the **Subject** and **Body** fields. You can also use internal NetSuite IDs. See Using Internal IDs.
	Include View Record Link	Include a link to the current workflow record in NetSuite at the end of the email. A link appears in the email with the text **View Record**. The recipient can click the link and log in to NetSuite to view the record.
Attachment	File	Select this option and browse for the NetSuite-supported files from the File Cabinet. See Attaching Files to an Email.
	From Field	Select a **Record (Join Field)** and a **Field** that references a **Document** field. See Attaching Files to an Email.
	Include Statement	Include a statement for workflows based on Customer, Lead, or Prospect record types. See Attaching Statements to an Email.
	Type	This option applies to the statement you are including and is unavailable if you are not including a statement with your email message.

Section	Property	Description
	Statement Date	Select a date for when the statement was generated. This option applies to the statement you are including and is unavailable if you are not including a statement with your email message.
	Start Date	Select a date for the earliest transactions you want to show on the statement. Leave this field blank if you want to show all transactions for this customer. This option applies to the statement you are including and is unavailable if you are not including a statement with your email message.
	Show Only Open Transactions	This option applies to the statement you are including and is unavailable if you are not including a statement with your email message.
	Consolidated Statement	If you use the Consolidated Payments feature, check the **Consolidated Statement** box to send a statement showing the overall balance for the customer-subcustomer hierarchy this customer is a part of. Clear this box to send a statement showing only the balance for this customer This option applies to the statement you are including and is unavailable if you are not including a statement with your email message.
	Use Customer's Locale	Select this option to generate the statement in the locale of the customer, instead of the default/company language. For a list of supported languages, see System Supported Languages in Configuring Multiple Languages. This option applies to the statement you are including and is unavailable if you are not including a statement with your email message.

Using Internal IDs

You can use internal NetSuite IDs to dynamically generate values in the **Cc**, **Bcc**, and **Body** fields on a Send Email action or in other similar fields in workflow actions.

For example, if the record in the workflow is a sales order, to generate the email as if it was sent by the sales rep for the customer, use the `{salesrep}` internal ID in the closing statement of the email in the **Body** field:

```
Thank you {entity}, for your order. We will notify you shortly regarding your shipping date.

Sincerely,
{salesrep}
```

You can also reference data from joined records. For example, to add the email for the sales rep to the above email, use `{salesrep.email}`.

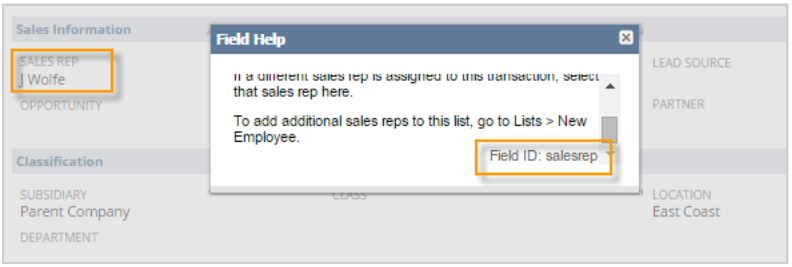
Finding the Internal NetSuite ID

Use the Field ID for a field in NetSuite to find the internal NetSuite ID. You can use a similar process to also find the internal ID for joined records.

To use field-level help to identify internal NetSuite IDs:

1. Go to Home > Set Preferences.
2. Enable **Show Internal IDs** and click **Save**.

3. In NetSuite, go to the record that contains the field that you want to reference.
4. Click the link for field-level help. The ID is at the bottom of the pop-up window that appears.



Attaching Files to an Email

Use the **Attachments** section for a Send Email action to attach a file. You can attach a single NetSuite-supported file, or you can attach a file contained in a Document custom field. The file size limit for attachments is 10MB.

You cannot select files from Image fields, only custom Document fields. In addition, to attach multiple files, create a .zip of all the files to send in the email and select the .zip file as the attachment.

To send a file as an attachment with the Send Email action:

1. If you have not already done so, create a Send Email action.

   See Creating an Action and Send Email Action.
2. To attach a file from a file record, under **Attachment** in the **Parameters** section, choose **File**.

   Click the double arrows and choose **List** to select from all files in the File Cabinet.

   Optionally, click **New** to create a new File record.

   Optionally, after you select a file, click **Open** to open the file record in NetSuite.
3. To attach a file from a Document field, under **Attachment** in the **Parameters** section, choose **From Field**.

   In the **Record** dropdown list, choose the record that contains the Document field and choose the field from the **Field** dropdown list.

Attaching Statements to an Email

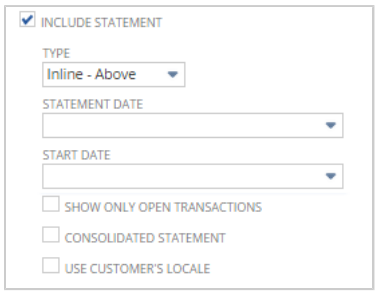
Use the **Attachments** section for a Send Email action to attach a statement. Use the **Include Statement** property to attach a statement for workflows based on the Customer, Lead, or Prospect record types.

For example, you can create a workflow that runs one time a week, uses a search for any customer more than 90 days overdue, and sends an email with the days overdue and attached statement.

To include a statement with the Send Email action:

1. If you have not already done so, create a Send Email action.

   See Creating an Action and Send Email Action.
2. Under **Attachment** in the **Parameters** section, enable **Include Statement**.



3. Specify the following properties:

Property	Description
Type	Rendering style of the statement. Choose one of the following options: ■ **Inline - Above.** The statement information appears above the body text of the email. ■ **Inline - Below.** The statement information appears below the body text of the email. ■ **Default.** The statement is attached based on the customer's email preference from the **Preferences** tab of the customer record. If the customer's email preference type is also set to **Default**, the statement is attached according to the email preference of the current user in the workflow. ■ **HTML.** The statement appears as an HTML attachment in the email. ■ **PDF.** The statement appears as a PDF attachment in the email.
Statement Date	Date the statement was generated.
Start Date	Date of the earliest transactions to show on this statement. Leave blank to show all transactions for the customer.
Show Only Open Transactions	Include only transactions in an open state. If there is no **Start Date**, the statement includes all open transactions in the customer history.
Consolidated Statement	Send a statement showing the overall balance for the customer-subcustomer hierarchy. Requires the Consolidated Payments feature. Disable this option to show only a balance.
Use Customer's Locale	Generate the statement in the language of the customer, instead of the default or company language. Only available for workflows based on the Customer record type.

Attaching Transactions to an Email

Use the **Include Transaction** property in **Attachments** section for a Send Email action to attach a the details of a transaction record. Attach transaction details for workflows based on any transaction record type.

For example, you can create a workflow that automatically sends an email when an order is placed in the Web store or sends an email with a packing slip whenever an order ships.

To include a transaction with the Send Email action:

1. If you have not already done so, create a Send Email action.

See Creating an Action and Send Email Action.

2. Under **Attachment** in the **Parameters** section, enable **Include Transaction**.
3. Specify one of the following options for the **Type** property:

Option	Description
Inline - Above	The transaction details appear above the body text of the email.
Inline - Below	The transaction details appear below the body text of the email.
Default	The transaction details are attached based on the customer's email preference from the **Preferences** tab of the customer record. If the customer's email preference type is also set to **Default**, the transaction details are attached according to the email preference of the current user in the workflow.
HTML	The transaction details appear as an HTML attachment in the email.
PDF	The transaction details appear as a PDF attachment in the email.

Set Field Display Label Action

Use the Set Field Display Label action to change the UI label of a field when the action executes. You can also use the Set Field Display Label action to change the UI label of a field in a sublist that SuiteFlow supports. The Set Field Display Label action is available at the record level, but cannot be applied to a sublist action group.

If the Multi-Language feature is enabled on your account, you can enter translations for the new label. For more information, see Entering translated strings for Set Field Display Labels.

Use the Before Record Load trigger to change the UI label when the record loads into the browser from editing, viewing, or creating a new instance of the base record type for the workflow.

Use a client trigger to change the UI label as the user interacts with the record form. For more information about executing the action dynamically if the record meets certain conditions, see Using Conditional Fields with Actions.

Set Field Display Label Action Properties

The following screenshot shows the **Parameters** section for the Set Field Display Label action:

The following table describes the Set Field Display Label action properties:

Property	Description
Field	Name of the field for the base record type and sub types for the workflow on which to change the label.
Label	String to use to replace the default UI label.

Note: For more information about adding actions to a workflow, including common action properties and conditions, see Action Conditions and Creating an Action.

Entering translated strings for Set Field Display Labels

Every company has a default language that is set by an Administrator or a user with the Set Up Company permission. The default company language can only be changed by an Administrator or a user with the Set Up Company permission. However, if the Multi-Language feature is enabled in your account, you can enter translations for button labels on the Workflow Action page regardless of the default company language.

You can add languages to your account on the General Preferences page. Any languages that you add to your account will be added to the translation table on the Workflow Action page. For more information, see the help topic Configuring Multiple Languages.

1. On the **Workflow Action** page, enter the translated label in the **Translation** column, next to the appropriate language.
2. If you do not want to enter a translation for a language, leave the **Translation** column next to the language empty. If you do not enter a translated label, the default company language will be used for the button label.
3. When you are done entering translated labels, click **Save**.

Set Field Display Type Action

Use the Set Field Display Type action to change the display type of a field when the action executes.

Use the Before Record Load trigger to change the display type when the record loads into the browser from editing, viewing, or creating a new instance of the base record type for the workflow.

Use a client trigger to change the display type as the user interacts with the record form. For more information about executing the action dynamically if the record meets certain conditions, see Using Conditional Fields with Actions.

For example, use this action in an approval workflow, to allow only the current approver in the workflow to set the approval field, either by disabling the field, setting it to inline, or by hiding it.

Set Field Display Type Action Properties

The following screenshot shows the **Parameters** section for the Set Field Display Type action:

The following table describes the Set Field Display Type action properties:

Property	Description
Field	Name of the field for the base record type and sub types for the workflow on which to change the display type.
Display Type	Display option for the field. Select one of the following options: ■ **Hidden.** Field cannot be seen on the record or transaction. ■ **Inline.** Used for informational purposes only. Cannot be edited. ■ **Disabled.** Field cannot be edited. ■ **Normal.** Field that can be edited. For more information about each of the display types, see the help topic Setting Display Options for Custom Fields.

Note: For more information about adding actions to a workflow, including common action properties and conditions, see Action Conditions and Creating an Action.

Set Field Mandatory Action

Use the Set Field Mandatory action to require a field in a workflow or sublist to have a value before a user can move to the next state in a workflow. Mandatory fields in NetSuite appear with an orange asterisk next to the field name. Also use this action to make fields that are required by default an optional field.

Use the Before Record Load trigger to make a field required when the record loads into the browser from editing, viewing, or creating a new instance of the base record type for the workflow.

Use a client trigger to make a field required as the user interacts with the record form. For more information about executing the action dynamically if the record meets certain conditions, see Using Conditional Fields with Actions.

Note: If you use the Set Field Mandatory action on a check box field, NetSuite does not make the field required on the record form when you save the record.

Set Field Mandatory Action Properties

The following screenshot shows the **Parameters** section for the Set Field Mandatory action:

The following table describes the Set Field Mandatory action properties:

Property	Description
Field	Name of the field for the base record type and sub types for the workflow to require a value for on the record form.
Mandatory	Make the field required. If the field was previously required, clear this box to make the field optional.

Note: For more information about adding actions to a workflow, including common action properties and conditions, see Action Conditions and Creating an Action.

Set Field Value Action

Use the Set Field Value action to set the value of a record field in a workflow instance. Use this action to set any field value on a record form including multi-select and date fields. Typically, this action is used to set the value of a field as a record enters a state.

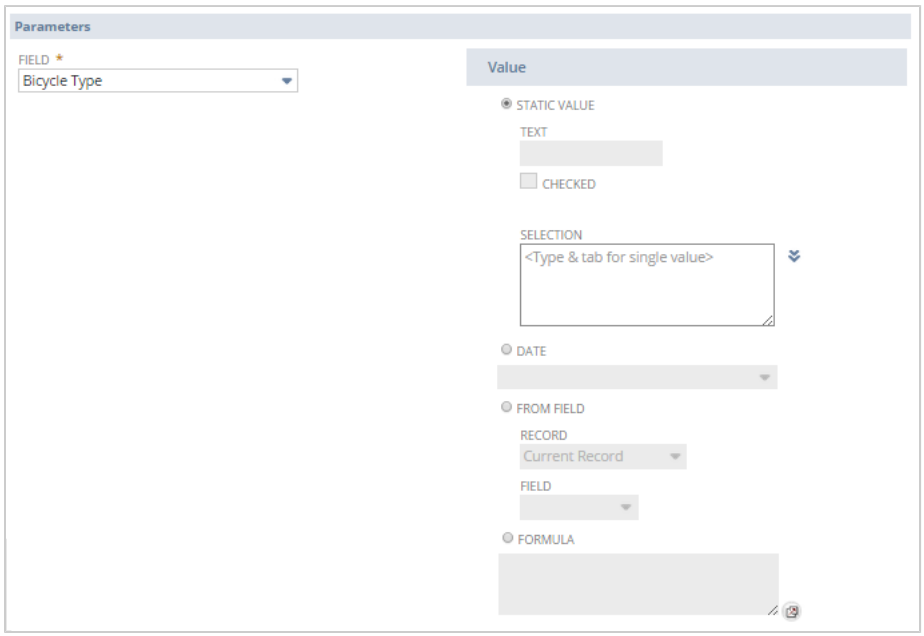
The triggers used with the Set Field Value action determines when the field value is set. The following table describes the valid triggers for a Set Field Value action:

Trigger	Description
Entry	Executes the first time a record enters a state. Default trigger for the Set Field Value action.
Exit	Executes the first time a record exits a state in a workflow.
Before Record Load	Executes before the record form loads into the browser. Cannot be used to set values on an existing record. See Set Field Value Action Guidelines.
Any client trigger (Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing, Before User Submit)	Executes as the user interacts with the record form.
Before Record Submit	Executes after a user clicks **Save** on a record, but before NetSuite saves the record to the database.
After Record Submit	Executes after NetSuite saves the record to the database.
Scheduled	Executes on a schedule. See Scheduling an Action.

Note: To execute the action dynamically if the record meets certain conditions, see Using Conditional Fields with Actions.

Set Field Value Action Properties

The following screenshot shows the **Parameters** section for the Set Field Value action that contains a multi-select field:



The following table describes the Set Field Value action properties:

Property	Description
Field	Name of the field for the base record type and sub types for the workflow on which to set the value on the record form.
Static Value	Set a specific value for the field. Set one of the following options: ■ **Text.** Use with field types such as free form fields. Enter the field value in the **Text** box. ■ **Checked.** Use with checkbox fields. ■ **Selection.** Click the double-arrows to select a value or values from the **List** popup. For multi-select fields, this option only applies for new records. See Set Field Value Action Guidelines.
Date	Select a date range. Applies to fields of type Date.
From Field	Sets the value of the field with a different field value. Set the following options: ■ **Record.** Select **Current Record** or another joined record type that contains the field value to use. ■ **Field.** Field value to use.
Formula	Use the Formula Builder to create a formula used to derive the field value. For client triggers, use a JavaScript or SuiteScript formula. For server triggers, use a SQL formula. See Defining a Condition with Formulas.

Note: For more information about adding actions to a workflow, including common action properties and conditions, see Action Conditions and Creating an Action.

Set Field Value Action Guidelines

Important: Use the following guidelines when working with the Set Field Value action.

- You cannot use a Before Record Load trigger to set field values for existing records. This limitation applies to the Set Field Value and Go To Record actions. Use a custom action to update fields on an existing record.
- The value you specify in a Set Field Value action on a Before Record Load trigger does not override a stored custom field or a field sourced from another record. To set this type of field, use a client trigger. The Before User Edit client trigger is the functional equivalent of the Before Record Load server trigger, except that Before User Edit executes in the client, after the record is loaded from the database. Records must be in edit mode to enable you to specify field values using the Set Field Value action; values are static in view mode.
- If you set the value of a multi-select field, the new values replace any original values in the field on the **Field** dropdown list.
- For multi-select fields, the **Date** and the **Formula** options are disabled. Only the **Static Value** and **From Field** options are available
- You can place a single field value instead of multiple values into a multi-select field.
- You cannot use the Set Field Value action to set the External ID of a record. External IDs are designed to be used as references in NetSuite. If setting the Internal ID is not sufficient for your business needs, you must create a custom field to set the External ID for a record.

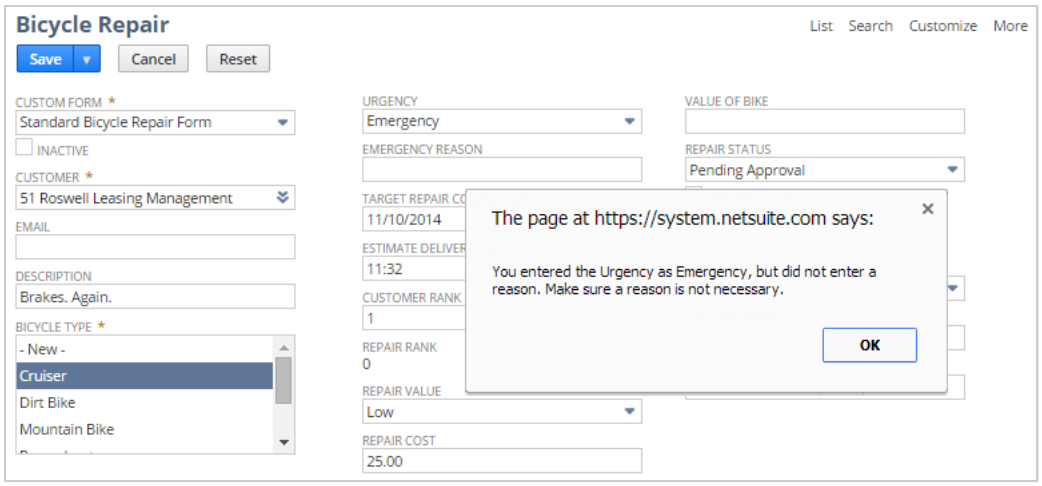
Show Message Action

Use the Show Message action to display a message to the user, such as when you want to validate values entered on the record form. The action displays the message in a popup window with an **OK** button. Unlike the Return User Error or Confirm actions, the Show Message action does not prevent subsequent user actions on the record form.

If the multi-language feature is enabled on your account, you can enter translations for the popup message. For more information, see Entering translated strings for popup messages.

The Show Message action is a client side action, it displays a message in the client browser as users interact with the record form. As such, you can only select client side triggers for the Show Message action. For example, if you want a message to pop up after a user enters an invalid value in a field, select the After Field Edit client side trigger. Additionally, on the Workflow Definition page, under Event Definition, set the Trigger Type to Before Record Load.

The following screenshot shows the execution of the Show Message action:



Show Message Action Properties

The following screenshot shows the **Parameters** section for the Show Message action:

The following table describes the Show Message action properties:

Property	Description
Text	Text to display in the browser popup when the action executes.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Entering translated strings for popup messages

Every company has a default language that is set by an Administrator or a user with the Set Up Company permission. The default company language can only be changed by an Administrator or a user with the Set Up Company permission. However, if the Multi-Language feature is enabled in your account, you can enter translations for button labels on the Workflow Action page regardless of the default company language.

You can add languages to your account on the General Preferences page. Any languages that you add to your account will be added to the translation table on the Workflow Action page. For more information, see the help topic Configuring Multiple Languages.

1. On the **Workflow Action** page, enter the translated label in the **Translation** column, next to the appropriate language.

2. If you do not want to enter a translation for a language, leave the **Translation** column next to the language empty. If you do not enter a translated label, the default company language will be used for the button label.
3. When you are done entering translated popup messages, click **Save**.

Subscribe To Record Action

Use the Subscribe To Record action to track a workflow custom instance or state field that contains a reference to a record. The Subscribe To Record action tracks the created record and identifies any change to the referenced record after the Subscribe To Record action executed. Create conditions for actions or transitions in the current workflow that depend on field values in the referenced record.

The Subscribe To Record action is designed to be used in conjunction with the Create Record action. For example, you create a Phone Call record in a workflow and then subscribe to the Phone Call record. You then create another action that executes only if the status of the Phone Call record changes.

The following table lists additional examples of working with the Subscribe To Record action:

Topic	Description
Creating and Subscribing to a Record	Example of creating and subscribing to a Phone Call record.
Lead Nurturing Workflow	Sample of a complete workflow that uses the Create Record and Subscribe To Record actions.
Creating and Using Workflow Fields, Creating and Using State Fields	Creating workflow and state fields to use with the Create Record and Subscribe To Record actions.

Subscribe To Record Action Properties

The following screenshot shows the **Parameters** section for the Subscribe To Record action:

The following table describes the Subscribe To Record action properties:

Property	Description
Field	Workflow field or workflow state field to use as a reference to a record.

Note: For more information about adding actions to a workflow, including common action properties and conditions, see Action Conditions and Creating an Action.

Subscribe To Record Action Guidelines

Important: Use the following guidelines when working with the Subscribe To Record action.

The Subscribe To Record action is not available on a Before Record Load trigger for a time entry record.

Transform Record Action

Use the Transform Record action to transform the data on a transaction record into another transaction record type. You can create workflows that processes transaction by transforming them into the next record type in a transaction record life cycle. For example, you can create a workflow to process sales orders and create invoices based on the appropriate date range, amounts, or other criteria.

Use the Transform Record action to transform the following types of records:

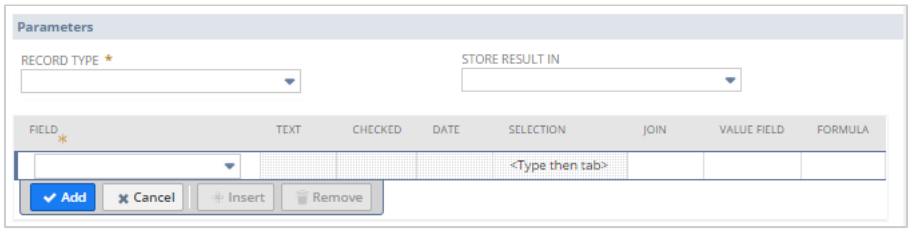
Assembly Build	Return Authorization
Cash Sale	Sales Order
Estimate	Vendor Bill
Invoice	Vendor Return Authorization
Opportunity	Work Order
Purchase Order	

When you transform a record, the base record type for the workflow does not change. For example, if you transform an Estimate into a Purchase Order, the record in the workflow is still an Estimate. You can use the Go To Record action to send the user to the newly created record, or create another workflow that uses a Transaction and Purchase Order as the base record type and sub type.

The functionality of the Transform Record action is the same as the SuiteScript 2.x method –record.transform(options).

Transform Record Action Properties

The following screenshot shows the **Parameters** section for the Transform Record action:



The following table describes the Transform Record action properties:

Property	Description
Record Type	Type of record to create by transforming the data in the current record for the workflow. The list of properties in the **Field** column changes depending on the type of record. Only supported record types for the current record in the workflow appear.
Store Result In	Workflow field or state field in which to store a reference to the created record. This field must exist before you create the action. Optional. Use this field to subscribe to a created record with the Subscribe To Record action. See Creating and Subscribing to a Record.
Field	Field values to populate when creating the record in the **Record Type** dropdown list. For each field, select the field name, set the appropriate properties in the other columns, and click **Add**. For more information, see Setting Field Values in Action Definitions.

Property	Description
	Note: These are not field values for the base record type of the workflow.

Note: For more information about adding actions to a workflow, including common action properties, see Creating an Action.

Transform Record Action Guidelines

Important: Use the following guidelines when working with the Transform Record action.

- The Transform Record action does not appear on the **New Action** window if you cannot transform the base record type for the workflow.
- The Transform Record action is not available on a Before Record Submit or After Record Submit trigger for a time entry record.
- If the Intercompany Cross-Subsidiary Fulfillment feature is enabled in your account, Inventory Location may become a required transformation parameter. If possible, use SuiteScript to transform Sales Orders to Item Fulfillments if the Intercompany Cross-Subsidiary Fulfillment feature is enabled in your account. SuiteScript will let you pass the inventorylocation transformation parameter. If SuiteScript is not an option and a workflow must be used, set Allow Cross Subsidiary Transaction to F and Inventory Location to blank for future transactions.

Triggers Reference

See the following topics for reference information about triggers in SuiteFlow:

- Server triggers. Server triggers occur when a record is read from or written to the NetSuite database or when a record enters or exits a state in a workflow. See Server Triggers Reference.
- Client triggers. Client triggers execute when a user interacts with a record form in NetSuite. See Client Triggers Reference.

More Information About Server and Client Triggers

Use the following table to get more information about working with server and client triggers:

Task	For more information
Understanding the SuiteFlow trigger execution model	SuiteFlow Trigger Execution Model
View which triggers did or did not execute	Workflow Execution Log
Using triggers for workflow initiation	Workflow Initiation
Using triggers for actions	Action Triggers
Using triggers for transitions	Transition Triggers

Workflow Triggers Quick Reference

The following table shows all actions and the triggers each action supports. The default trigger action is identified by the letter D.

	On Entry On Exit	Before Record Load	Before Record Submit	After Record Submit	Before User Edit	Before Field Edit	After Field Edit	After Field Sourcing	Before User Submit	Scheduled
Add Button Action	X	D	—	—	—	—	—	—	—	—
Confirm Action	—	—	—	—	—	X	—	—	D	—
Create Line Action	D	X	X	X	—	—	—	—	—	X
Create Record Action	D	X	X	X	—	—	—	—	—	X
Custom Action	D	X	X	X	—	—	—	—	—	X
Go To Page Action	X	—	—	D	—	—	—	—	—	—
Go To Record Action	X	—	—	D	—	—	—	—	—	—
Initiate Workflow Action	D	X	X	X	—	—	—	—	—	X
Lock Record Action	X	D	—	—	—	—	—	—	—	—
Remove Button Action	D	X	—	—	—	—	—	—	—	—
Return User Error Action	X	X	D	—	X	X	X	X	X	—
Send Campaign Email Action	D	—	—	X	—	—	—	—	—	X
Send Email Action	D	—	—	X	—	—	—	—	—	X
Set Field Display Label Action	X	D	—	—	X	X	X	X	—	—
Set Field Display Type Action	X	D	—	—	X	X	X	X	—	—
Set Field Mandatory Action	X	D	—	—	X	X	X	X	—	—
Set Field Value Action	D	X	X	X	X	X	X	X	X	X
Show Message Action	—	—	—	—	D	X	X	X	X	—
Subscribe To Record Action	D	X	X	X	—	—	—	—	—	X
Transform Record Action	X	—	—	X	—	—	—	—	—	X

D indicates the default trigger action.

X indicates that the trigger action is supported.

The following table shows actions in groups and the triggers each action group supports. The default trigger action is identified by the letter D.

	On Entry On Exit	Before Record Load	Before Record Submit	After Record Submit	Scheduled
Add Button Action	X	D	—	—	—
Create Line Action	D	X	X	X	X

	On Entry On Exit	Before Record Load	Before Record Submit	After Record Submit	Scheduled
Create Record Action	D	X	X	X	X
Go To Page Action	X	—	—	D	—
Go To Record Action	X	—	—	D	—
Initiate Workflow Action	D	X	X	X	X
Lock Record Action	X	D	—	—	—
Remove Button Action	D	X	—	—	—
Return User Error Action	X	X	D	—	—
Send Campaign Email Action	D	—	—	X	X
Send Email Action	D	—	—	X	X
Set Field Display Label Action	X	D	—	—	—
Set Field Display Type Action	X	D	—	—	—
Set Field Mandatory Action	X	D	—	—	—
Set Field Value Action	D	X	X	X	X
Subscribe To Record Action	D	X	X	X	X
Transform Record Action	X	—	—	X	X

D indicates the default trigger action.

X indicates that the trigger action is supported.

The following table shows actions in the sublist action group and the triggers each sublist action group supports. The default trigger action is identified by the letter D.

	On Entry On Exit	Before Record Load	Before Record Submit	After Record Submit	Scheduled
Create Record Action	D	X	X	X	X
Return User Error Action	X	X	D	—	—
Send Email Action	D	—	—	X	X

	On Entry On Exit	Before Record Load	Before Record Submit	After Record Submit	Scheduled
Set Field Value Action	D	X	X	X	X
D indicates the default trigger action. X indicates that the trigger action is supported.					

Server Triggers Reference

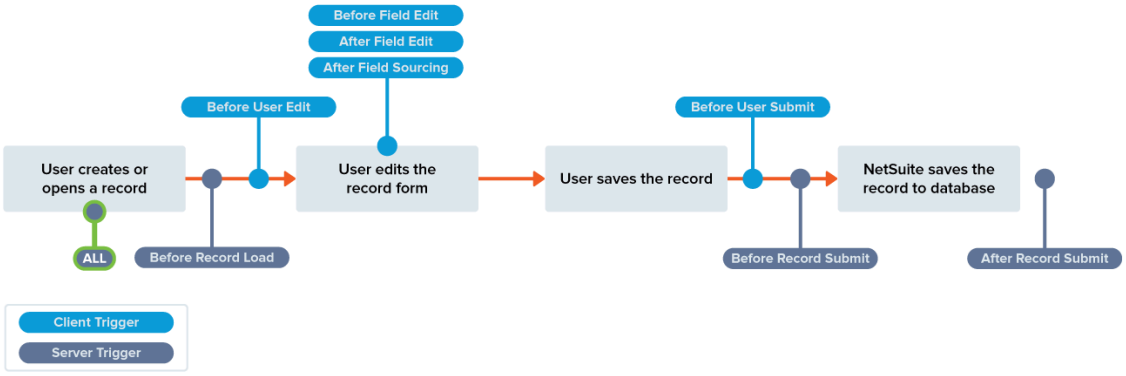
The following table describes the server triggers and their functionality. See Workflow Triggers and Server Triggers for general information about server triggers.

Trigger Type	Applies To	Description
All	▪ Workflow initiation	Initiates a workflow instance for any triggering event for a workflow. See All Trigger.
Entry	▪ Actions ▪ Transitions	Executes the first time a record enters a state in a workflow. See Entry Trigger.
Before Record Load	▪ Workflow initiation ▪ Actions ▪ Transitions	Executes before a record loads into the browser. See Before Record Load Trigger.
Before Record Submit	▪ Workflow initiation ▪ Actions ▪ Transitions	Executes after a user clicks **Save** on a record, but before NetSuite saves the data to the database. See Before User Submit Trigger.
After Record Submit	▪ Workflow initiation ▪ Actions ▪ Transitions	Executes after NetSuite saves the record data to the database. See After Record Submit Trigger.
Exit	▪ Actions	Executes the first time a record exits a state in a workflow. See Exit Trigger.
Scheduled	▪ Workflow initiation ▪ Actions ▪ Transitions	Directs NetSuite to initiate a workflow or execute an action or transition on a schedule. See Scheduled Trigger.

All Trigger

The All trigger type is a server trigger that starts an instance of a workflow for any triggering event that occurs for a record. For example, when a user views a record, a workflow instance initiates because a Before Record Load event executes on the viewing of a record.

The following diagram shows typical record events and when the All trigger executes:



Guidelines for the All Server Trigger

Important: Use the following guidelines when using the All server trigger.

- Use the All trigger when the functionality of the workflow is not dependent on when a workflow may initiate and enter the entry state.
- In general, you should choose a more specific trigger type for workflow initiation. Consider when a workflow should initiate and enter the entry state. For example, if it is necessary for a workflow to only initiate after a record is saved to the database, choose the After Record Submit trigger instead.
- When you cancel a workflow instance for a single record, NetSuite reloads the record in the NetSuite UI. Therefore, for workflow that initiate **On Update** and use the All or Before Record Load server triggers, the workflow instance re-initiates when you cancel it. To avoid this result, use the Before Record Submit or After Record Submit server triggers on workflows set to initiate **On Update**.

Entry Trigger

The Entry trigger is a server trigger and executes the first time a record enters a state in a workflow. Use the Entry trigger if you want an action or transition to execute only when a record enters a state for the first time. The Entry trigger applies to all actions except the Confirm and Show Message actions and to all transitions.

The actions and transitions that execute on the Entry trigger depend on the server trigger on which the record entered the state. If the record enters the state on a Before Record Load trigger, only actions and transitions that may execute on the Before Record Load trigger will execute. You can control the trigger on which a record enters a state by using the correct trigger on the transition into the state.

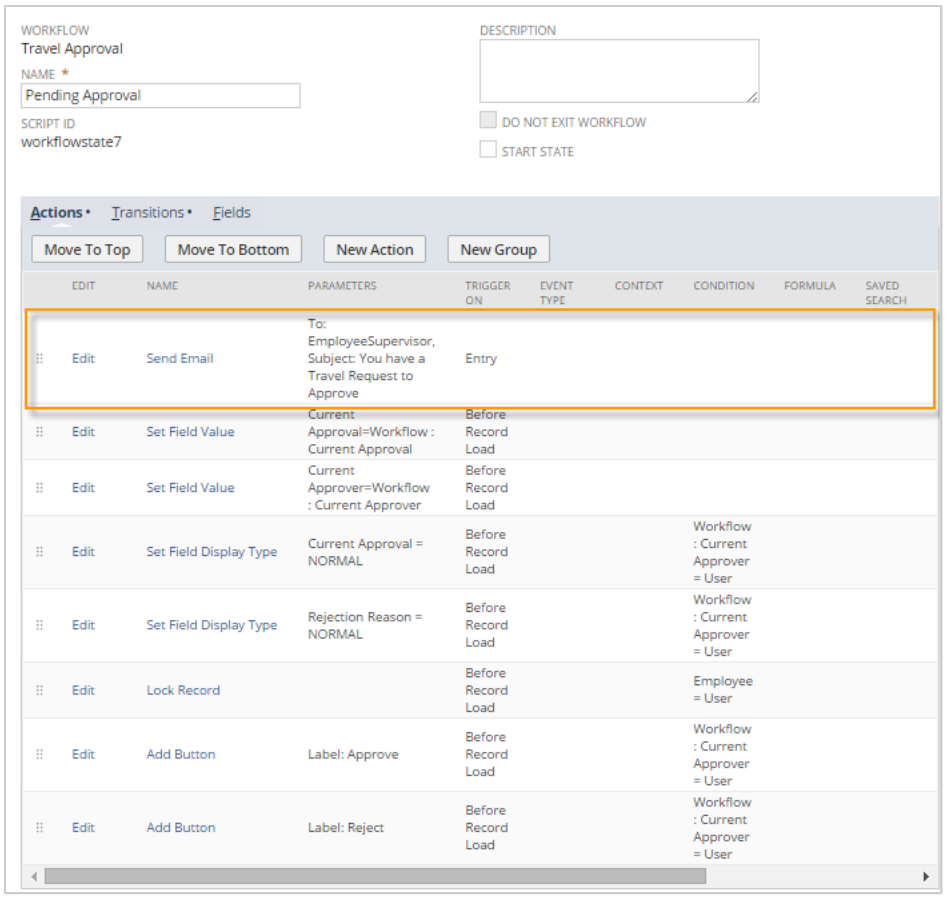
For example, a state contains a Send Email action set to execute on the Entry trigger and the record enters the state on the Before Record Load server trigger. However, the Before Record Load trigger is not a valid trigger type for the Send Email action. Consequently, the Send Email action does not execute.

If you use the After Record Submit trigger on the transition into the state, any Send Email action set to trigger on Entry executes, but only the first time the record enters the state. For more information, see Rules and Guidelines for Workflow Triggers and Trigger Execution.

For a list of the valid actions for the Entry trigger, see Workflow Triggers Quick Reference.

Entry Trigger Example

The following state shows actions set to execute on the Entry and the Before Record Load triggers:



Trigger Type	Workflow Layout
Entry	The Send Email action executes when the record enters the state for the first time. If the record enters the state a second time, the Send Email action does not execute again.
Before Record Load	All other actions execute on Before Record Load, or each time the records loads when the record is in the Pending Approval state.

This state is designed so that after a Travel Request is created, the record enters the Pending Approval state on an After Record Submit, and NetSuite sends the email notifying the supervisor. The email only needs to be sent one time, not multiple times. Consequently, the Send Email action is set to execute on the Entry trigger.

Guidelines for Using the Entry Trigger

Important: Use the following guidelines when using the Entry trigger.

If you want an action or transition to execute every time a record enters a state, use a more specific trigger to identify when the action or transition should execute.

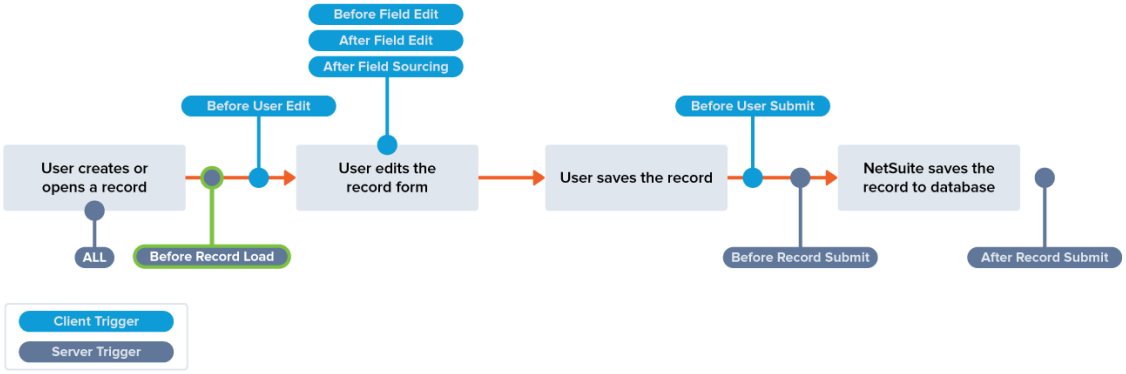
Before Record Load Trigger

The Before Record Load trigger is a server trigger and executes before a record loads into the browser. Actions and transitions set to execute on Before Record Load execute on the NetSuite server, after a user clicks **Edit** or **View**, and before the record loads and the user has access to the record.

The Before Record Load trigger can be used on the following elements of a workflow:

- Workflow initiation. See Workflow Initiation Triggers and SuiteFlow Trigger Execution Model.
- Actions. For a list of actions that support the Before Record Load trigger, see Workflow Triggers Quick Reference.
- Transitions. See Transition Triggers.

The following diagram shows typical record events and when the Before Record Load trigger executes:

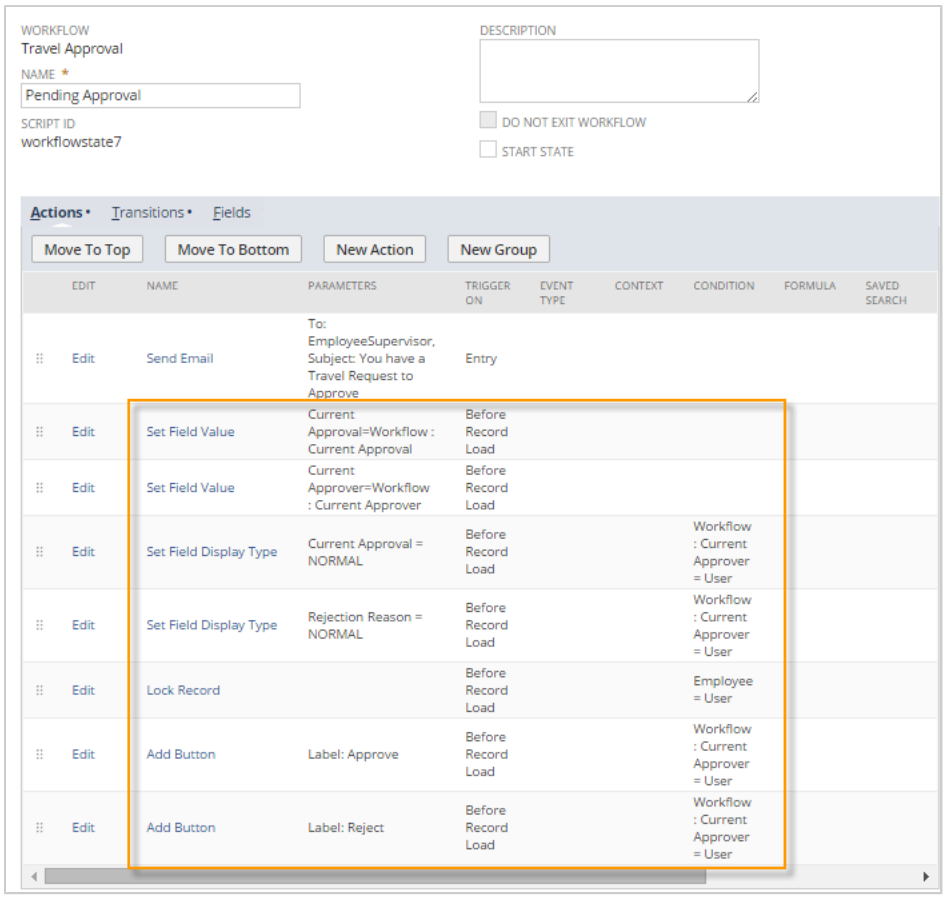


In general, use the Before Record Load trigger when you want to use a workflow to make changes to the record before the record loads into the browser. For example, use the Before Record Load trigger in the following situations:

- Add or remove buttons, depending on the user role or other conditions.
- Perform calculations based on field values and display those values when the records loads.
- Lock the record to make the record uneditable for specific users or user roles.
- Change the display type for fields to make certain fields required or hide them.
- Validate the record or check the record for permissions to prevent a record from being viewed.

Before Record Load Trigger Example

The following screenshot shows actions for a state set to execute on the Before Record Load and Entry triggers:



Trigger Type	Action Execution
Before Record Load	Several actions may execute before the record loads into the browser, depending on the user that accesses the record: ■ The two Set Field Value actions execute for any user. ■ The two Set Field Display Type actions and the two Add Button actions execute, only if the **Current Approver** field for the Travel Request is the user accessing the record. ■ The Lock Record action executes if the user accessing the record is the employee who submitted the Travel Request.
Entry	The Send Email action executes when the record enters the state for the first time.

This state is designed so that whenever the Travel Request record gets accessed, the approval buttons are added for the approver of the Travel Request and **Current Approval** and **Rejection Reason** fields become editable.

For the employee that submitted the Travel Request, the record is only locked, the buttons do not appear, and the **Current Approval** and **Rejection Reason** fields appear but are not editable.

Guidelines for Using the Before Record Load Trigger

Important: Use the following guidelines when using the Before Record Load trigger.

- Refreshing the browser reloads a record. If a user refreshes the browser during editing or viewing of a record in a state, all Before Record Load actions in that state execute again.

- You cannot set field values on an existing record if the Set Field Value actions execute on a Before Record Load trigger. Use the Set Field Value action with a Before Record Load trigger for new records or for setting workflow fields on new or existing records. Use a custom action to update fields on an existing record.
- The Before Record Load and Before User Edit triggers appear to be functionally similar. However, actions that use the Before Record Load trigger execute on the server, whereas actions that use the Before User Edit trigger execute in the browser.

  In addition, actions set to trigger on Before Record Load execute every time the record is loaded. Actions set to trigger on Before User Edit only execute right before the user edits the record. If an action modifies a record on a Before User Edit trigger, the edits are lost unless the user saves the record.

  The following table lists the action types that can be performed on each trigger and might need to be performed before the record appears to a user:

Type of Action	Before Record Load	Before User Edit
Set Field Value (new records)	Yes	Yes
Set Field Value (existing records)	No	Yes
Set Field Value (workflow fields)	Yes	No
Set Field Display Type (Disabled, Hidden, Normal)	Yes	Yes
Set Field Display Type (Inline)	Yes	Yes
Set Field Mandatory	Yes	Yes
Set Field Display Label	Yes	Yes
Add Button, Remove Button	Yes	No
Lock Record	Yes	No

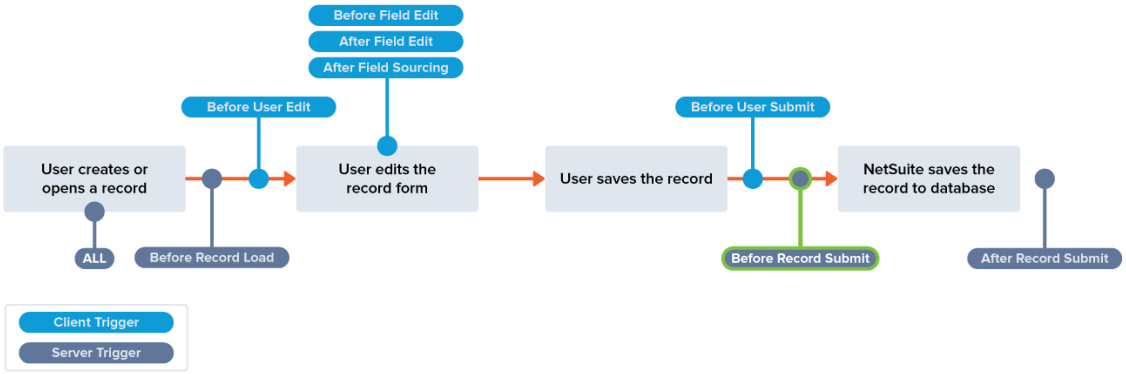
Before Record Submit Trigger

The Before Record Submit trigger is a server trigger and executes after a user clicks **Save** on a record, but before NetSuite saves the data to the database. Actions and transition set to execute on Before Record Submit execute on the NetSuite server, before the record data is 'submitted' to the database.

The Before Record Submit trigger can be used for the following elements of a workflow:

- Workflow initiation. See Workflow Initiation Triggers and SuiteFlow Trigger Execution Model. For an example, see Before Record Submit Trigger Example.
- Actions. For a list of actions that support the Before Record Submit trigger, see Workflow Triggers Quick Reference.
- Transitions. See Transition Triggers.

The following diagram shows typical record events and when the Before Record Submit trigger executes:



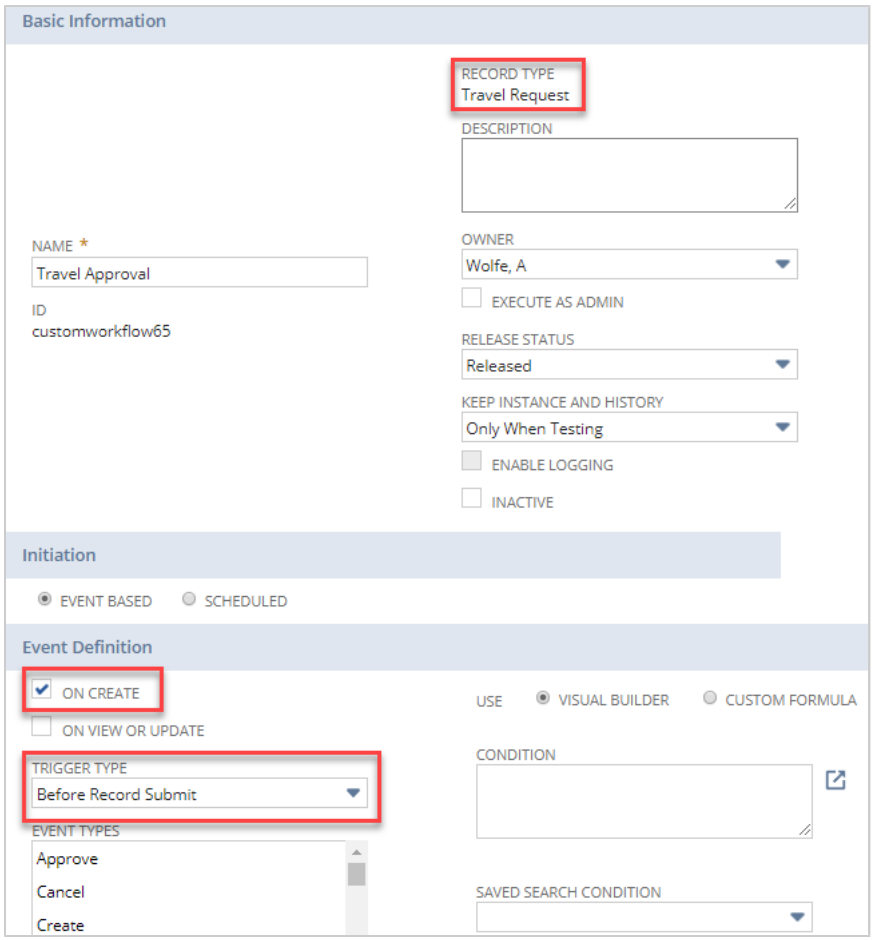
In general, use the Before Record Submit trigger when you want to use a workflow to make changes to the record before NetSuite saves the data to the database. For example, use the Before Record Submit trigger in the following situations:

- Validate record data before NetSuite submits it to the database.
- Perform permission checks or validate restrictions before NetSuite writes the data to the database. Use a Return Error Action on a Before Record Submit trigger to prevent a user from saving the record until the user correctly modifies the record.
- Set certain field values for a record, based on other field values entered by the user.

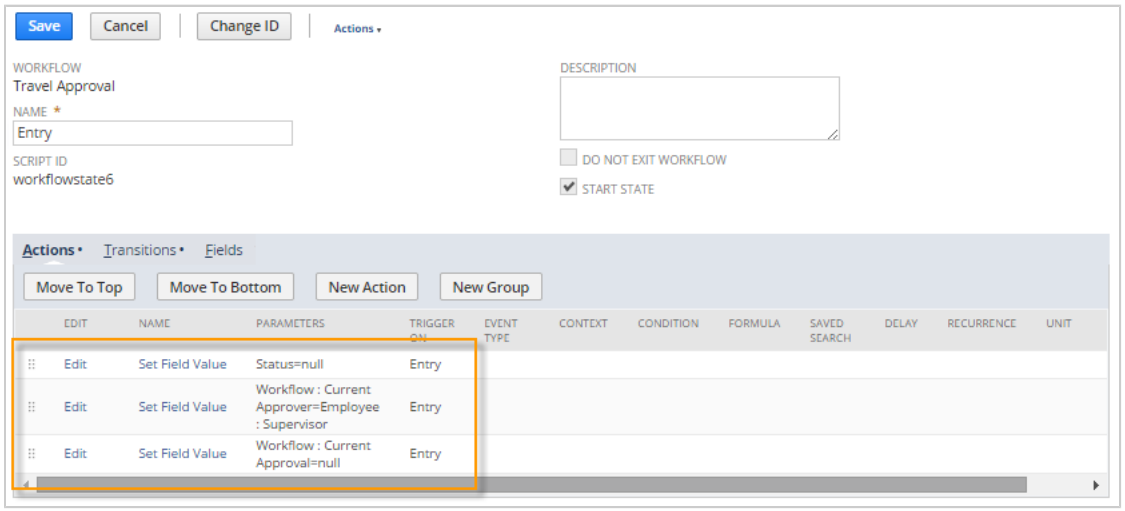
Before Record Submit Trigger Example

The following screenshots show the workflow definition and entry state for a workflow that initiates after a user clicks **Save** on a new Travel Request record:

Workflow Definition



Entry State



User Behavior	Workflow Behavior
User creates a new Travel Request record.	None.

User Behavior	Workflow Behavior
User clicks **Save** on the Travel Request record.	Workflow instance initiates, enters the entry state, executes the three Set Field Value actions. In this state, a transition executes on an After Record Submit trigger. Consequently, the record transitions to the next state after the record data is saved to the database.

The state is designed so the actions execute on the Entry trigger. They do not use the Before Record Load trigger, for example, because the workflow instance did not initiate until the Before Record Submit trigger executed. If the actions were set to execute on the Before Record Load trigger, they would not execute, because the record entered the state on a Before Record Submit trigger. For more information, see Rules and Guidelines for Workflow Triggers and Trigger Execution.

Before Record Submit Trigger Guidelines

Important: Use the following guidelines when using the Before Record Submit trigger.

- The Before Record Submit and Before User Submit triggers appear to be functionally similar. However, actions that use the Before Record Submit trigger execute on the server, whereas actions that use the Before User Submit trigger execute in the browser.

  For example, a Return User Error on a Before Record Submit trigger would execute on the server. The user would return to the record and all previously entered data would be lost. If a Return User Error or Show Message action executes on a Before User Submit trigger instead, the user clicks **OK** on the message and continues editing the record with no loss of data.
- In addition to the Before User Submit trigger, the Before Record Submit trigger also appears similar to the After Record Submit trigger, since they both execute after the user clicks **Save** on a record. However, for the Set Field Value, Custom, Create Record, Initiate Workflow, and Subscribe To Record actions, other differences may impact the workflow. See Comparing the Before Record Submit and After Record Submit Triggers.

After Record Submit Trigger

The After Record Submit trigger is a server trigger and executes after NetSuite saves the record data to the database. Actions and transitions set to execute on After Record Submit execute on the NetSuite server, after the record data is saved to the database.

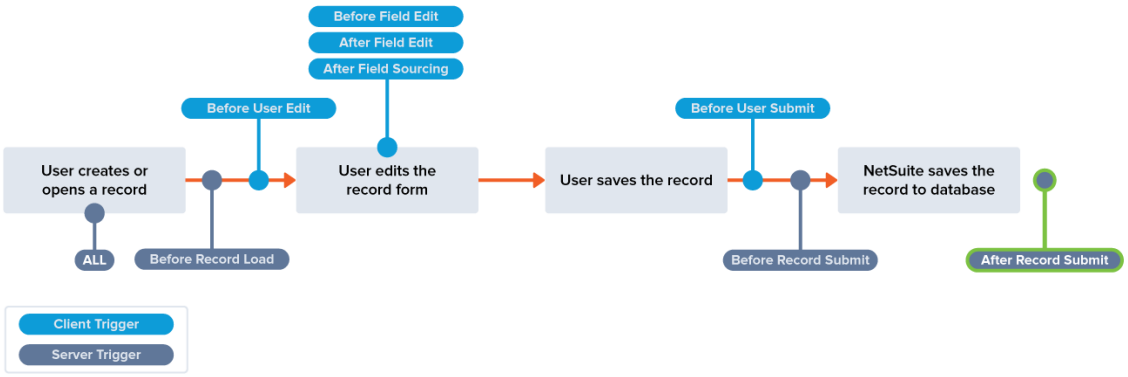
Note: In System Notes, NetSuite logs the role for the saved record as Administrator, even for non-Administrator users.

When an action executes on an After Record Submit trigger, NetSuite internally reloads the record, executes the action, and then saves the record again. NetSuite only saves the record again if the record has changed by the action.

The After Record Submit trigger can be used for the following elements of a workflow:

- Workflow initiation. See Workflow Initiation Triggers and SuiteFlow Trigger Execution Model.
- Actions. For a list of actions that support the After Record Submit trigger, see Workflow Triggers Quick Reference.
- Transitions. See Transition Triggers.

The following diagram shows typical record events and when the After Record Submit trigger executes:

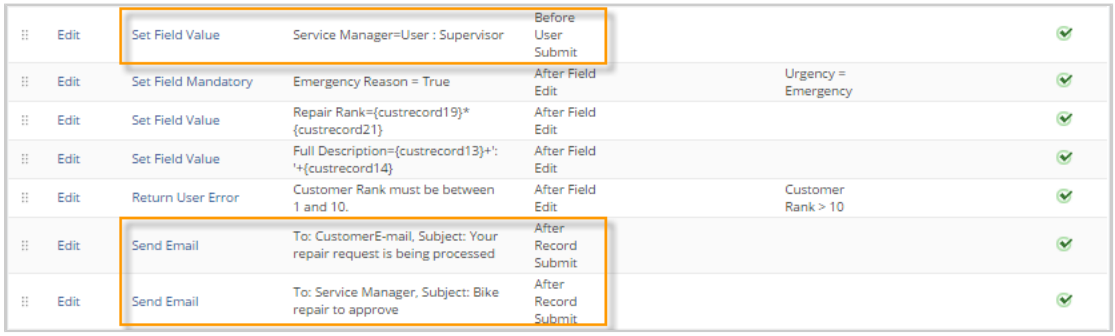


In general, use the After Record Submit trigger when you want to use a workflow to make changes to the record after NetSuite saves the data to the database. For example, use the After Record Submit trigger in the following situations:

- A field or field value value is not available until after NetSuite saves the record.
- Actions should occur after NetSuite saves the record. These actions can include the Send Email, Go To Page or Go To Record, or Create Record actions.
- Record must transition to another state after NetSuite saves the record.
- Some fields on the current record are set by NetSuite by other fields that are changed when the record is submitted.
- You want to perform other actions on an After Record Submit trigger that require you do computation and then store the result in the current record.

After Record Submit Trigger Example

The following screenshot shows actions for a state set to execute on the Before User Submit and After Record Submit triggers. The actions on the After Record Submit trigger require specific field values to be present in the database to execute.



Trigger Type	Action Behavior
Before User Submit	Sets the **Service Manager** field on the record, which is required for one of the Send Email actions.
After Record Submit	NetSuite sends both emails after NetSuite saves the record data to the database.

Trigger Type	Action Behavior
	The first Send Email action requires the Customer information from the record. The required data is present in the database after the record save. The second Send Email action requires the **Service Manager** field value, which is set before the record save by the Set Field Value action.

Comparing the Before Record Submit and After Record Submit Triggers

Although actions set to execute on the Before Record Submit and After Record Submit triggers both occur after the user clicks **Save** on a record, their use can affect edits made to the record.

The following table describes the differences between the two triggers for the Set Field Value, Custom, Create Record, Initiate Workflow, and Subscribe To Record actions:

Before Record Submit	After Record Submit	Other Considerations
All change made to the record by the workflow and the user are stored at the same time.	Any change made on the record by the workflow causes the record to be saved again when the workflow completes.	The Before Record Submit trigger results in better performance, for actions such as Set Field Value, as the record does not need to be potentially saved twice.
If the workflow execution fails for any reason, NetSuite cancels the entire operation. Any edit made by user is lost.	If the workflow execution fails for any reason, the data entered by the user has already been saved. The only changes that are not saved are those made by the actions that execute on the After Record Submit trigger.	Use the Before Record Submit trigger for any actions, such as Set Field Value, that depend on record consistency. Modifications to the record made on the After Record Submit trigger are rolled back in case of failure, which may leave the record in an inconsistent state. The inconsistency results from necessary record changes not being made.
Any business logic that executes when the record is stored has not executed yet, and can cause an inconsistent state of the record.	Any business logic that executes when the record is stored has already executed, resulting in a more consistent state for the record.	Typically, actions that use both trigger types modify the record but also may need to read the record, usually in an action condition. In this case, the After Record Submit trigger may be more effective.

Note: Most other actions can use either of the above triggers, but not both.

Exit Trigger

The Exit trigger is a server trigger and executes the first time a record exits a state in a workflow. Use the Exit trigger if you want an action to execute only when a record exits a state for the first time. The Exit trigger applies to all actions except the Confirm and Show Message actions.

The actions and transitions that execute on the Exit trigger depend on the server trigger on which the record exits the state. If the record exits the state on an After Record Submit trigger, only actions and transitions that may execute on the After Record Submit trigger will execute. You can control the trigger on which a record exits a state by using the correct trigger on the transition from the current state to another state.

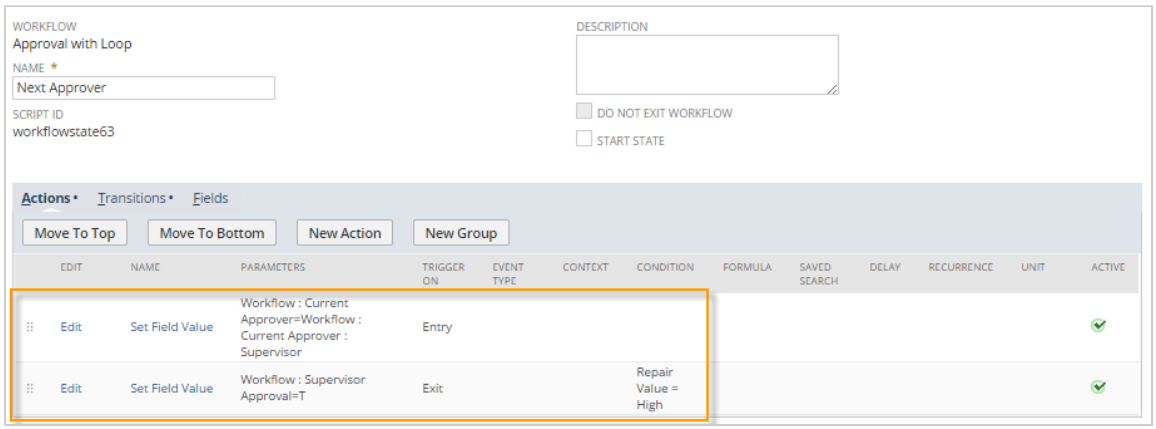
For example, a state contains a Lock Record action set to execute on the Exit trigger and the record exits the state on the After Record Submit server trigger. However, the After Record Submit trigger

is not a valid trigger type for the Lock Record action. Consequently, the Lock Record action does not execute. However, if you use the Before Record Load trigger on the transition from the state, any Lock Record action set to trigger on Exits executes, but only the first time the record exits the state. For more information, see Rules and Guidelines for Workflow Triggers and Trigger Execution.

For a list of the valid actions for the Exit trigger, see Workflow Triggers Quick Reference.

Exit Trigger Example

The following state shows actions set to execute on the Entry and Exit triggers:



Trigger Type	Workflow Behavior
Entry	The Set Field Value action executes when the record enters the state for the first time and sets the **Current Approver** field to the supervisor for the current approver.
Exit	The Set Field Value action executes when the record exits the state and sets the **Supervisor Approval** flag on the record to **True** if the **Repair Value** is set to **High**.

Exit Trigger Guidelines

Important: Use the following guidelines when using the Exit trigger.

In some cases, you can use the Entry and Exit triggers for the same purpose. If you require an action to only execute one time in a workflow, setting it to execute on Exit of one state or on Entry of a target state accomplishes the same result.

Scheduled Trigger

The Scheduled trigger is available for workflow initiation, transitions, and some actions. The Scheduled trigger directs NetSuite to initiate a workflow or execute an action or transition on a schedule. Set a schedule for workflow initiation or an action or transition execution by setting the **Trigger On** property to **Scheduled** and setting any additional schedule or condition requirements.

NetSuite runs a backend scheduler that processes schedules every 30 minutes. Workflow scheduled tasks include workflows scheduled to initiate or actions and transitions scheduled to execute. When the scheduler finds a workflow task that meets the schedule and all conditions associated with the task, NetSuite executes the task.

The method by which a scheduled workflow task gets executed depends on the type of task:

Workflow Task Type	Execution Method
Workflow initiation	When you set the schedule for workflow initiation, set the workflow status to Released, and then save the workflow definition. Every 30 minutes, the scheduler analyzes the schedule requirements and conditions for workflow initiation. If a workflow definition meets the schedule and condition requirements, the scheduler runs the saved search and initiates an instance of the workflow on each record returned by the saved search. For more information, see Initiating a Workflow on a Schedule and Scheduling a Workflow.
Action	A scheduled action is considered for execution when a record in a workflow instance stays in a state with a scheduled action. Every 30 minutes, the scheduler analyzes the schedule properties for such actions. If an action meets the schedule properties and any additional conditions on the action, the action executes on the workflow instance and record. For more information, see Scheduling an Action and Action Conditions.
Transition	A scheduled transition is considered for execution when a record in a workflow instance stays in a state with a scheduled transition. Every 30 minutes, the scheduler analyzes the schedule properties for such transitions. If the transition meets the schedule properties and any additional condition requirements, the transition executes on the workflow instance and record. See Scheduling a Transition and Transition Conditions.

Note: The backend scheduler that processes scheduled tasks runs every 30 minutes, but depending on the number of scheduled tasks, a particular task may be evaluated, or considered for execution, later. Similarly, after a task is evaluated for execution, the execution can be delayed due to number of tasks being processed by NetSuite. Consequently, a task scheduled for exact time may not be executed exactly at the specified time. The execution may occur at a later point. In addition, a task will never execute before the scheduled time

Time Zones for the Scheduled Trigger

When you select the date and time to run a scheduled task, the task runs in the time zone of the NetSuite user that creates the schedule. Whenever a user defines a new schedule or edits an existing schedule, the dates and times used in the schedule are saved in the time zone of the current user. Even if the user later changes the time zone preference, without changing the schedule, the schedule keeps the original date and time, using the original time zone.

Note: NetSuite does not consider the time zone of the workflow owner or the user who initiated a workflow with a scheduled action or transition.

Scheduled Trigger Guidelines

Important: Use the following guidelines when using the Scheduled trigger.

- Scheduled workflows, actions, or transitions do not execute if the **Release Status** of the workflow is not set to **Released**. See Release Status.

- For list of supported actions for the Scheduled trigger, see Workflow Triggers Quick Reference.
- Scheduled actions do not execute in an exit state after a record enters the exit state. The record exits the workflow before the scheduled action executes. See Exit States.

Client Triggers Reference

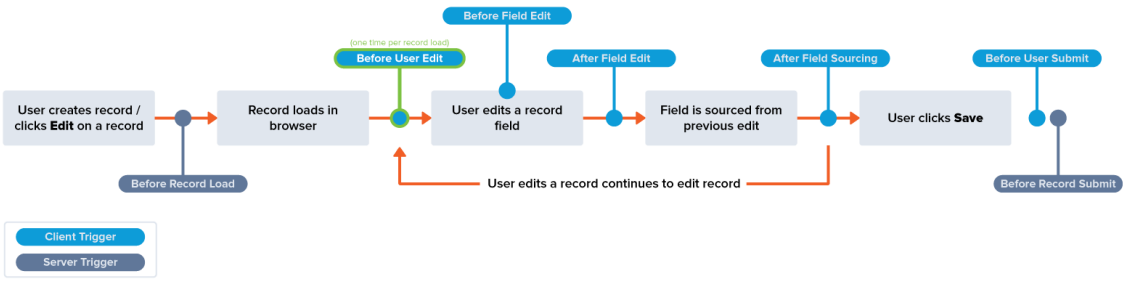
The following table describes the client triggers and their functionality. See Workflow Triggers and Client Triggers for general information about client triggers.

Trigger Name	Description
Before User Edit	Executes after a record form loads into the browser and before the user makes any changes to the form. See Before User Edit Trigger.
Before Field Edit	Executes before a user completes changes to a field on a record form. See Before Field Edit Trigger.
After Field Edit	Executes after a user completes changes to a field on a record form. See After Field Edit Trigger.
After Field Sourcing	Executes after all dependent field values on a record form have been populated. See After Field Sourcing Trigger.
Before User Submit	Executes after a user clicks **Save** on a record and before the Before Record Submit server trigger executes. See Before User Submit Trigger.

Before User Edit Trigger

The Before User Edit trigger is a client trigger and executes after a record form loads into the browser and before the user makes any changes to the form. Before User Edit is a client trigger because it occurs after the form loads into the browser. The trigger occurs due to user edit of an existing record or creation of a new record. Before User Edit only occurs one time when the record form is present in the browser.

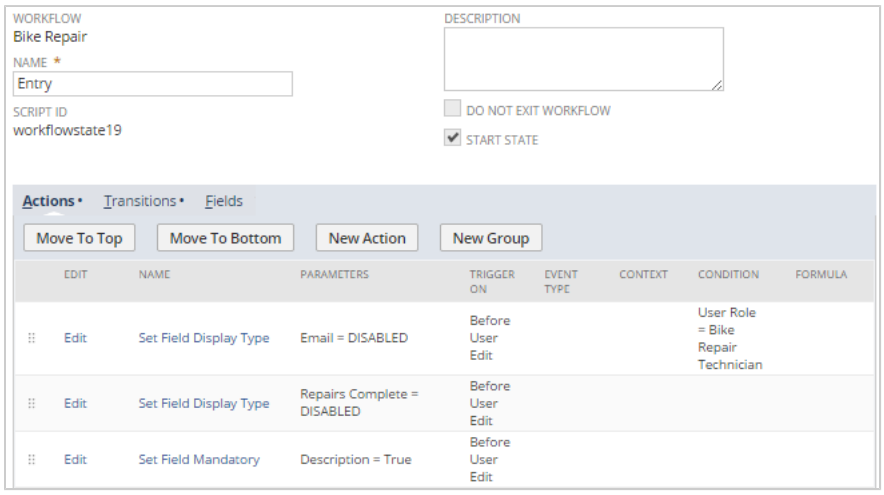
The following diagram shows common record edit events and when the Before User Edit trigger executes:



Use this trigger to make changes to the record form in the browser before the user edits any field. You can use this trigger with any of the field value actions, such as the Set Field Value action, or with the Return User Error action. For a list of all the actions that support this trigger, see Workflow Triggers Quick Reference.

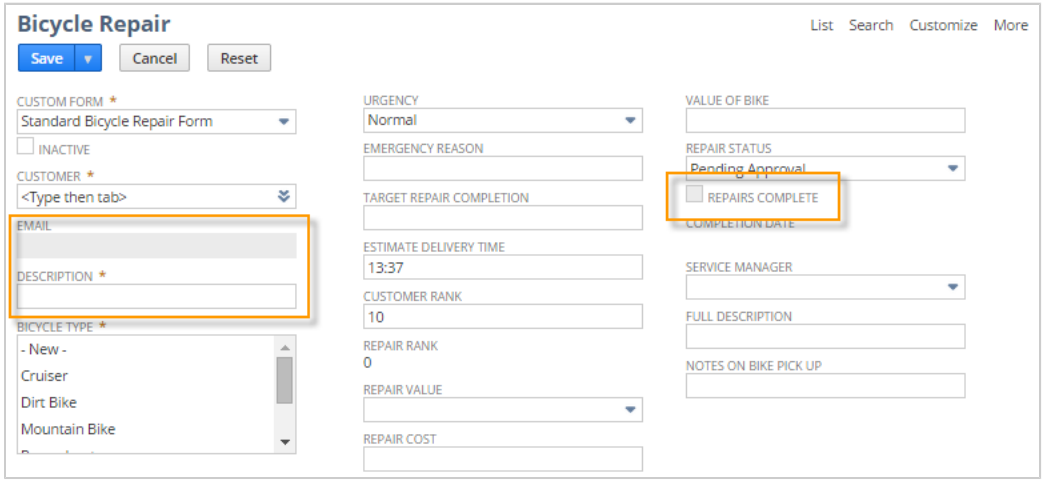
Before User Edit Trigger Example

The following screenshot shows a state with actions set to execute on the Before User Edit trigger:



Action Type	Action Execution
Set Field Display Type	The two Set Field Display Type actions disable the **Email** field and the **Repairs Complete** checkbox, respectively. The fields are visible but are not editable.
Set Field Mandatory	On a Bike Repair record, the **Description** field is not required by default. The Set Field Mandatory makes the **Description** field required.

This state is designed to customize the record form, depending on the user role accessing the record in the workflow. The following screenshot shows the respective action results on the record form:



Before User Edit Trigger Guidelines

Important: Use the following guidelines when using the Before User Edit trigger.

- For SuiteScript users, the Before User Edit trigger is the workflow equivalent of the **pageInit** client event. For JavaScript, execution of the Before User Edit trigger is similar to the **onLoad** client event.
- If a user reloads a record during the time that the record is in a state that includes actions that execute on the Before User Edit trigger, the actions execute again.

- The Before Record Load and Before User Edit triggers appear to be functionally similar. However, actions that use the Before Record Load trigger execute on the server, whereas actions that use the Before User Edit trigger execute in the browser.

  In general, you should use the Before Record Load trigger over the Before User Edit trigger. The server-side processing of actions is typically safer than client-side processing because there is a greater assurance of consistency in behavior. There can be inconsistent behavior if something unexpected occurs in the client environment, such as a browser upgrade.

  The following table lists the action types that can be performed on each trigger and might need to be performed before the record appears to a user:

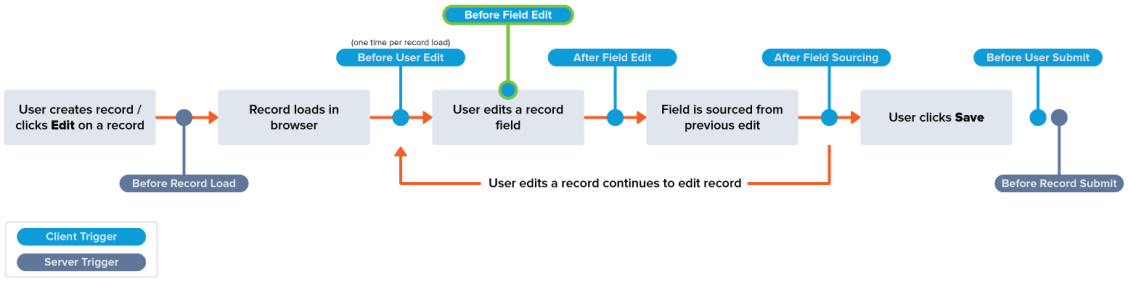
Type of Action	Before Record Load	Before User Edit
Set Field Value (new records)	Yes	Yes
Set Field Value (existing records)	No	Yes
Set Field Display Type (Disabled, Hidden, Normal)	Yes	Yes
Set Field Display Type (Inline)	Yes	No
Set Field Mandatory	Yes	Yes
Set Field Display Label	Yes	No
Add Button, Remove Button	Yes	No
Lock Record	Yes	No

Before Field Edit Trigger

The Before Field Edit trigger is a client trigger and executes before a user completes changes to a field on a record form. Before Field Edit is a client trigger because it occurs during the edit of a record during the time that it is loaded in the browser. Before Field Edit only occurs during the edit of a field on a record form when the record is in a state.

When you use the Before Field Edit trigger for an action, you must also identify the name of the client field in the **Client Fields** multi-select box for the action. This identifies which record field to execute the action on. For more information about **Client Fields** for actions, see Creating an Action and Using Conditional Fields with Actions.

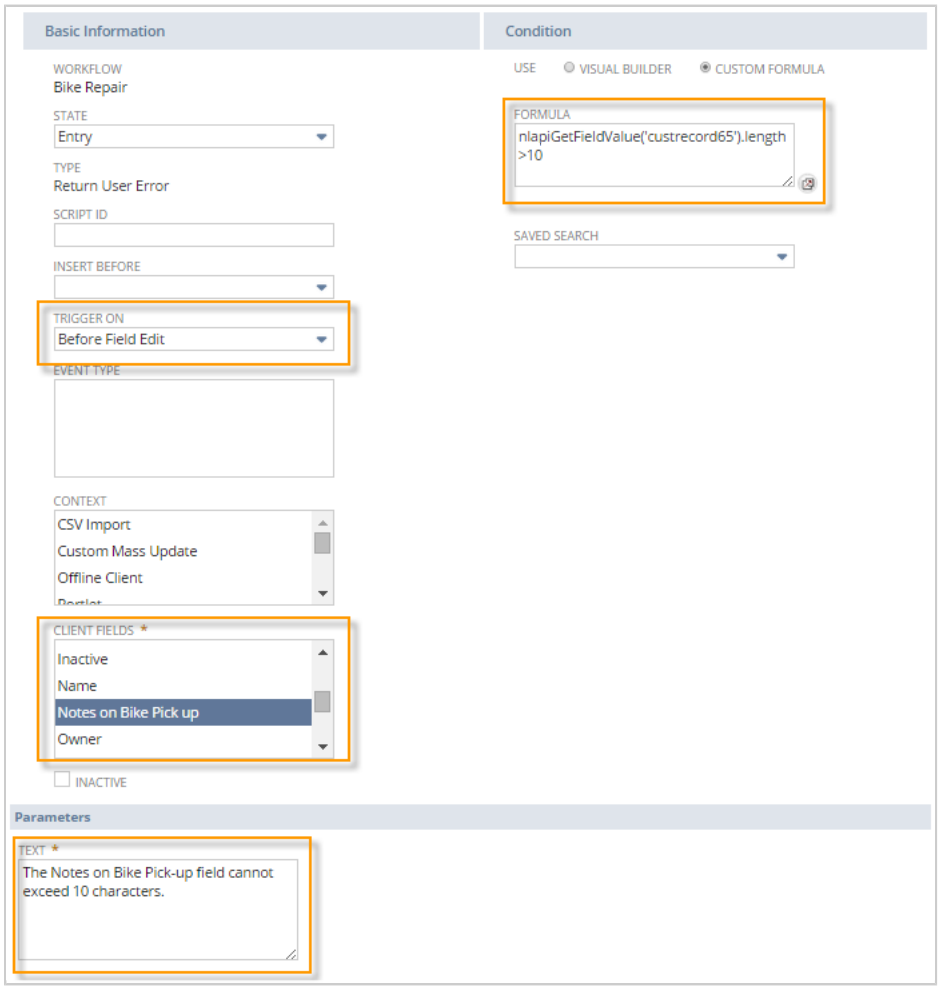
The following diagram shows common record edit events and when the Before Field Edit trigger executes:



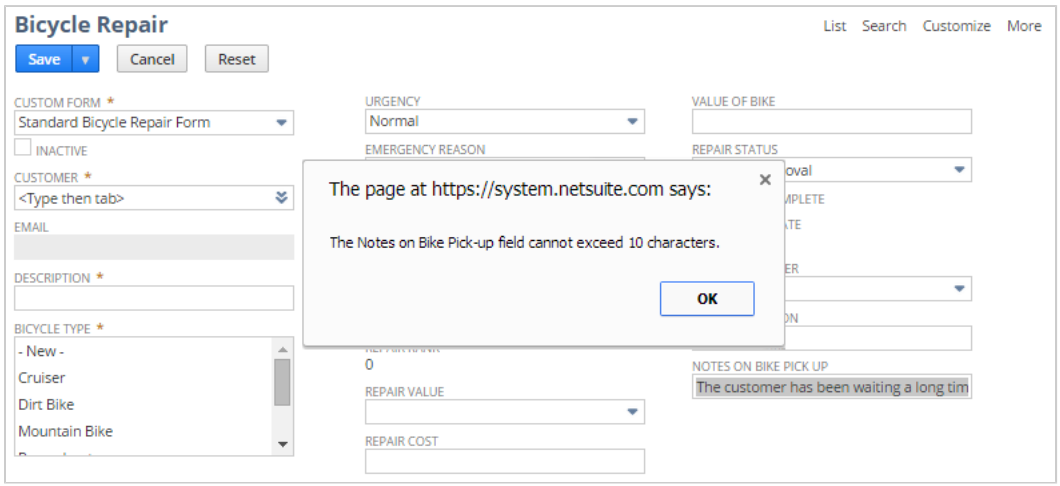
Use this trigger to validate field lengths, restrict field entries to a predefined format, or validate a field against values in an associated field. For example, use this trigger with a Return User Error action to alert users that they must fix a field value before the user continues editing the form. For a list of all the actions that support this trigger, see Workflow Triggers Quick Reference.

Before Field Edit Trigger Example

The following screenshot shows a Return User Error action used to validate the value of the **Notes on Bike Pick-up** field on a Before Field Edit trigger:



The following screenshot shows the results of the Return User Error action on the record form. The error appears after the user moves away from the **Notes on Bike Pick-up**. Clicking **OK** returns the window focus to the field that needs fixed. The popup keeps appearing until the user fixes the error.



Before Field Edit Trigger Guidelines

Consider the following when you use the Before Field Edit trigger:

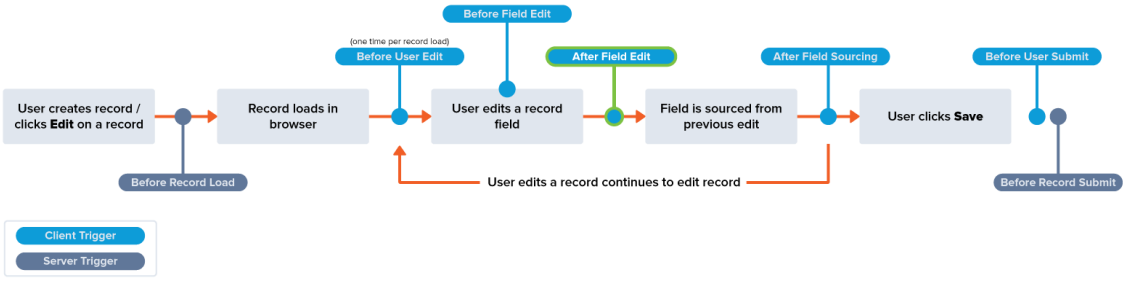
- For SuiteScript users, the Before Field Edit trigger is the workflow equivalent of the **validateField** client event. For JavaScript, the Before Field Edit trigger execution is the equivalent of an **onBlur** client event.

After Field Edit Trigger

The After Field Edit trigger is a client trigger and executes after a user completes changes to a field on a record form. After Field Edit is a client trigger because it occurs during the edit of a record during the time that it is loaded in the browser. After User Edit only occurs after the edit of a field on a record form when the record is in a state.

When you use the After Field Edit trigger for an action, you must also identify the name of the client field in the **Client Fields** multi-select box for the action. This identifies the fields that, if changed, change the values of other fields. For more information about using the **Client Fields** property on actions, see Creating an Action and Using Conditional Fields with Actions.

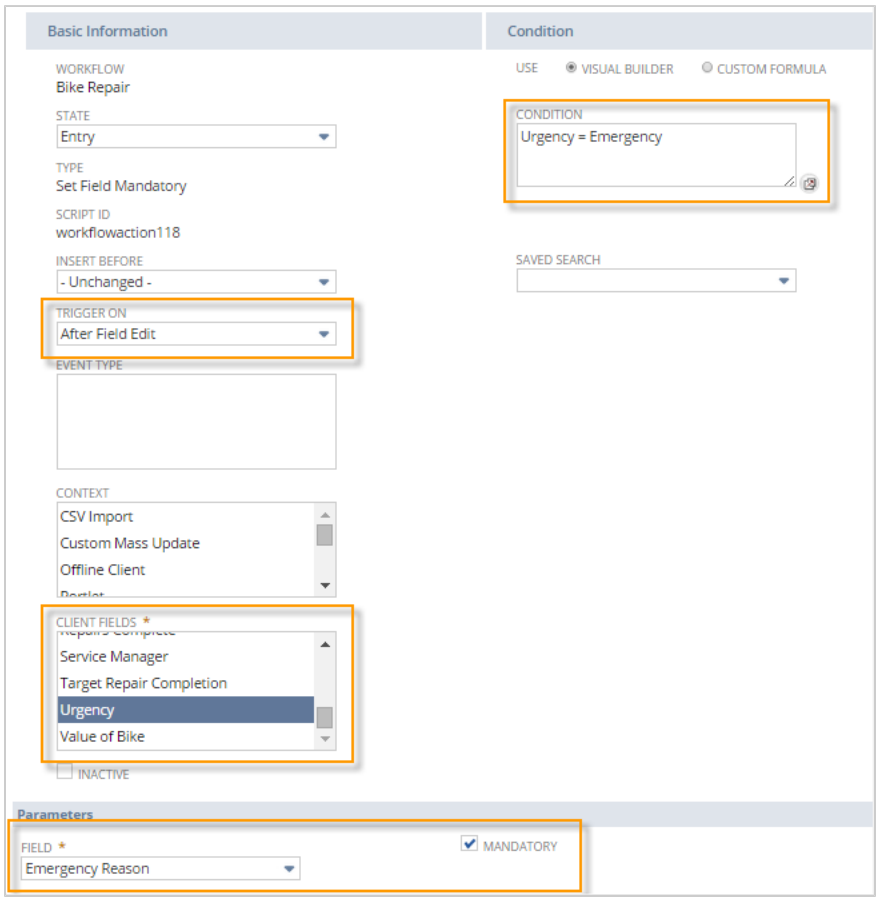
The following diagram shows common record edit events and when the After Field Edit trigger executes:



Use this trigger to set field values based on the values of other fields changed by the user. For a list of all the actions that support this trigger, see Workflow Triggers Quick Reference.

After Field Edit Trigger Example

The following screenshot shows a Set Field Mandatory action used to set the **Emergency Reason** field to required if the value selected for the **Urgency** field is **Emergency**:



After Field Edit Trigger Guidelines

Important: Use the following guidelines when using the After Field Edit trigger.

- For SuiteScript users, the After Field Edit trigger is the workflow equivalent of the **fieldChanged** client event. For JavaScript, actions on the After Field Edit trigger execute similar to an **onChange** JavaScript client event. As one field changes, so does different field.
- Actions on an After Field Edit trigger can also occur in response to an action on a Before Record Load trigger. For example, an After Field Edit action is set to execute whenever field A changes, and an action on a Before Record Load trigger changes the value of field A. The After Field Edit action executes in response to the change of field A.

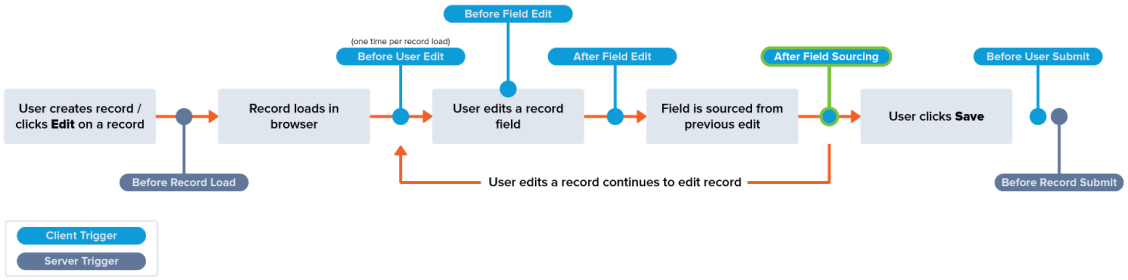
After Field Sourcing Trigger

The After Field Sourcing trigger is a client trigger and executes after all dependent field values on a record form have been populated. The fields can be populated by dependencies on other fields in the same record or in joined records. After Field Edit is a client trigger because it occurs during the edit of a record during the time that it is loaded in the browser.

For example, a user specifies a **Sales Rep** on a record and the custom **Sales Rep Email** field is populated with the sales rep email from the sales rep record. The custom field definition specifies the sourcing relationship between the **Sales Rep** and **Sales Rep Email** fields. The After Field Sourcing trigger occurs after the **Sales Rep Email** field is populated.

When you use the After Field Sourcing trigger for an action, you must also identify the name of the client field in the **Client Fields** multi-select box for the action. This identifies the fields that, if populated, populate other fields. For more information about using the **Client Fields** property on actions, see Creating an Action and Using Conditional Fields with Actions.

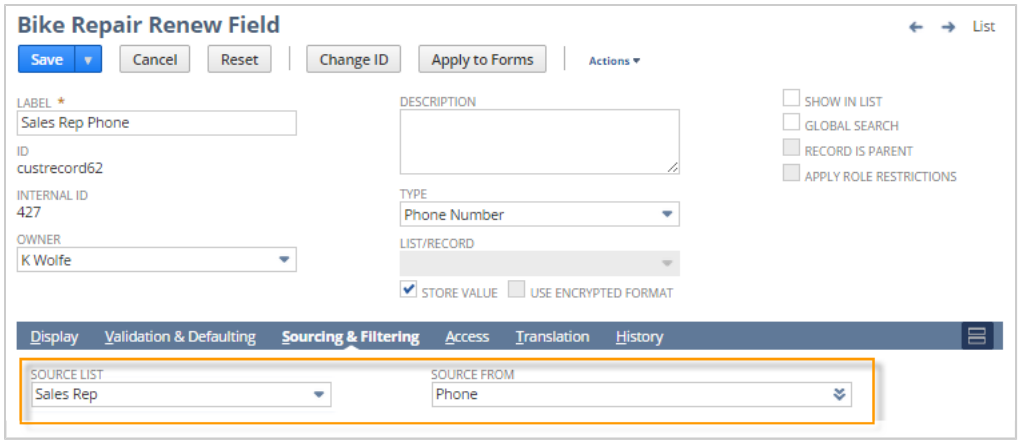
The following diagram shows common record edit events and when the After Field Sourcing trigger executes:



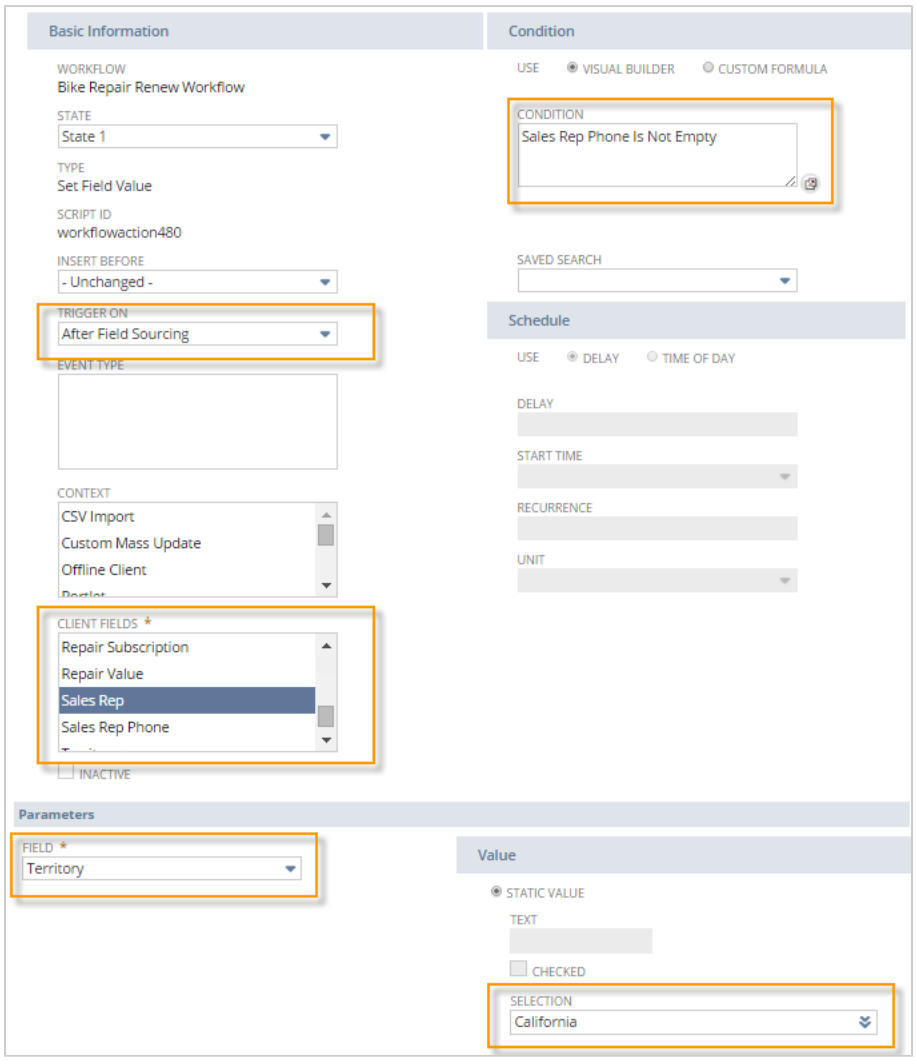
Use this trigger when you want to set field values based on other sourced values or you want to limit the users who specify field values on a record, even if the fields must contain values. For a list of all the actions that support this trigger, see Workflow Triggers Quick Reference.

After Field Sourcing Example

The following screenshot shows a custom field definition with the **Sales Rep Phone** sourced from the **Phone** field on the Sales Rep record:



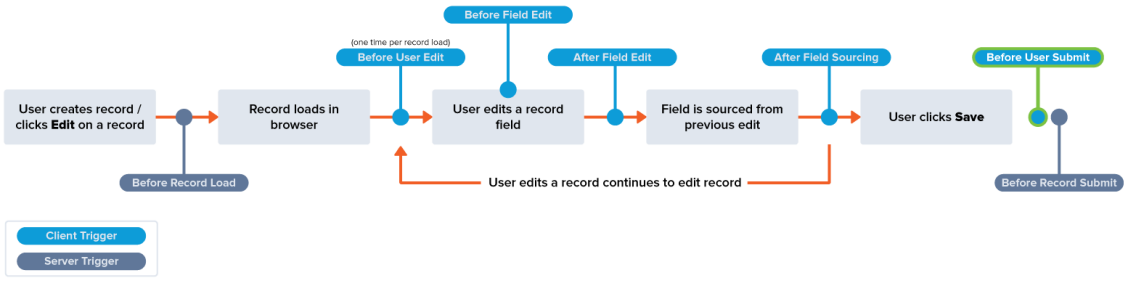
The following screenshot show a Set Field Value action that executes on the After Field Sourcing trigger to set another field, **Territory**, after the **Sales Rep** field is entered by the user on the record:



Before User Submit Trigger

The Before User Submit trigger is a client trigger and executes after a user clicks **Save** on a record and before the Before Record Submit server trigger executes. Before User Submit is a client trigger because it occurs when the form is present in the browser.

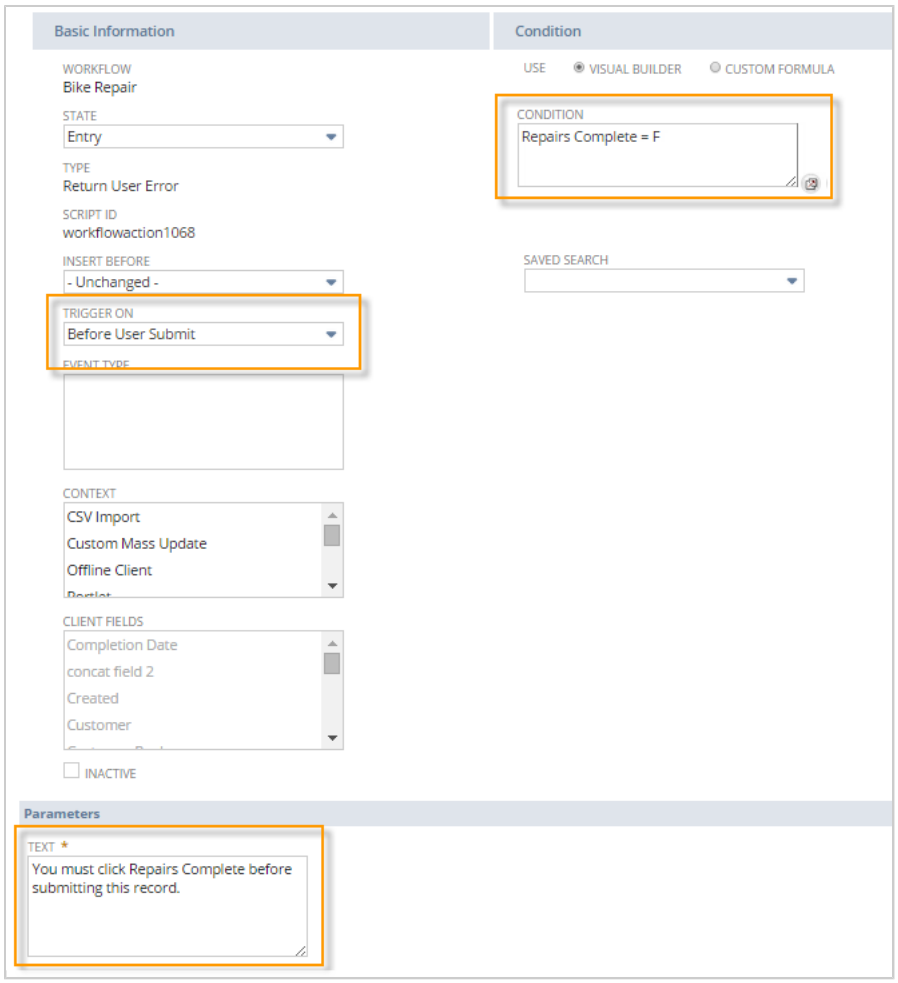
The following diagram shows common record edit events and when the Before User Submit trigger executes:



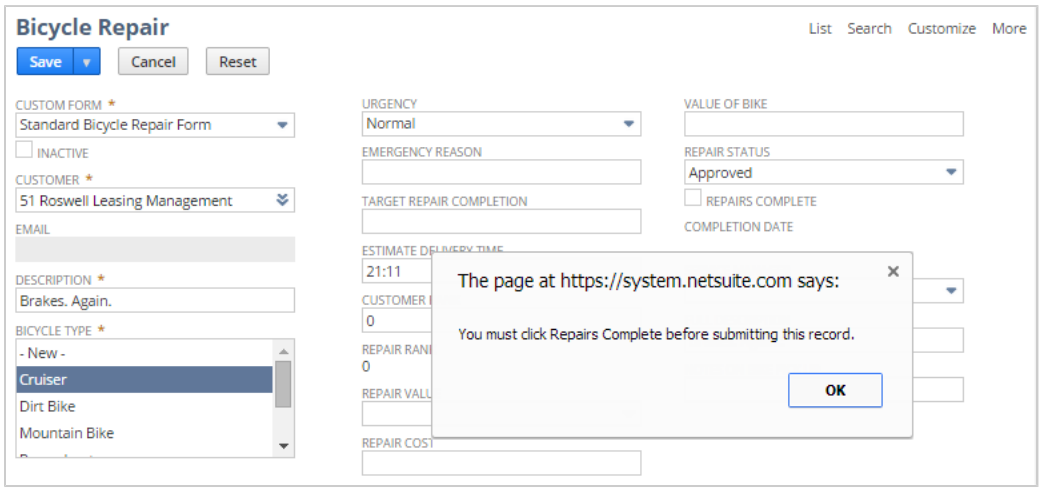
Use this trigger to validate a record form before the Before Record Submit trigger executes. Using this trigger with the Return User Error action, instead of the Before Record Submit trigger, enables a user to return to the form to fix any field value before submitting the form to the NetSuite database. For a list of all the actions that support this trigger, see Workflow Triggers Quick Reference.

Before User Submit Trigger Example

The following screenshot shows a Return User Error action on the Before User Submit trigger that validates that the **Repairs Complete** checkbox is selected before the user saves the record:



If the user clicks **Save** on the record without selecting the **Repairs Complete** checkbox, NetSuite displays the error message. The user clicks **OK** and then selects the checkbox.



Before User Submit Trigger Guidelines

Important: Use the following guidelines when using the Before User Submit trigger.

- The Before Record Submit and Before User Submit triggers appear to be functionally similar. However, actions that use the Before Record Submit trigger execute on the server, whereas actions that use the Before User Submit trigger execute in the browser.

  For example, a Return User Error on a Before Record Submit trigger would execute on the server. The user would return to the record and all previously entered data would be lost. If a Return User Error or Show Message action was used on a Before User Submit trigger instead, the user clicks **OK** on the message and continues editing the record with no loss of data.

States Reference

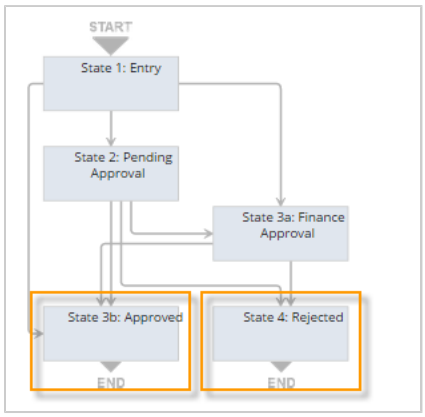
Use the following topics for reference information when working with exit states in a workflow:

- Exit States
- Non-Exiting Workflow States

Exit States

An exit state in a workflow is the final state the record enters before a workflow completes. Depending on the layout of a workflow, a single workflow may have multiple exit states. The diagrammer designates any state without a transition to another state as an **End**, or exit, state.

The following screenshot shows a workflow with two exit states, **State 3b Approved** and **State 4 Rejected**:



The diagrammer removes the **End** state designation when you create a transition from an exit state to another state or you enable the **Do Not Exit Workflow** property on the state.

Trigger Execution in Exit States

NetSuite only executes the actions for the server trigger on which a record entered an exit state and then completes the workflow. Subsequent server triggers do not execute.

For example, if a record enters an exit state on a Before Record Load trigger, only actions set to trigger on Entry or Before Record Load execute. Then, the record exits the workflow. The Before Record Submit and After Record Submit triggers do not execute. For more information about the order of trigger execution in a state, see SuiteFlow Trigger Execution Model.

You can use one of the following methods to control which actions may execute in an exit state:

- Enable the **Do Not Exit Workflow** property for the state. Consequently, the record stays in the state and the workflow never completes.
- Create a new state as the exit state. On the transition into the new exit state, set the trigger for the transition to a trigger that allows the actions in the exit state to execute. For example, if you have actions that trigger on After Record Submit, make sure the transition into the state uses the Before Record Submit or After Record Submit trigger.

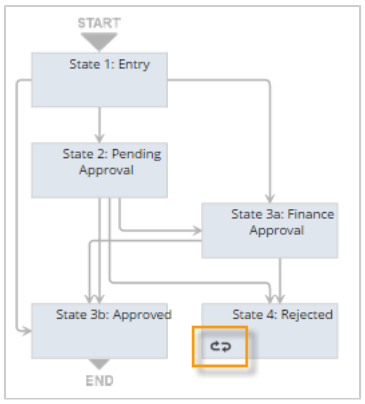
Exit State Guidelines

Important: Use the following guidelines when working with exit states.

Scheduled actions do not execute in the exit state. The workflow completes and leaves the exit state before scheduled actions can execute. See Scheduling an Action.

Non-Exiting Workflow States

Use non-exiting states to execute a workflow indefinitely. You can set a state that has no transitions as a non-exiting state with the **Do Not Exit Workflow** property. The diagrammer shows non-exiting states with a circular arrows icon:



Note: If you create a transition from the state, the **Do Not Exit Workflow** property is unchecked. In addition, the property cannot be set if a state has any transitions.

Use non-exiting states in the following types of workflows:

- Single state workflows. Use the state to execute an action or actions any time a record of a specific type is created or edited.
- Approval-based workflows. For example, after a request has been rejected, set the record to enter a non-exiting state with a Lock Record action. If a user or specific group of users accesses the record, the record cannot be edited.

Event Types Reference

The following table lists all event types, the server trigger for which they appear, whether they apply to workflows that initiate **On Create** or **On Update**, and any limitations:

Event Type	Trigger Types (Workflow Initiation)	Limitations
Approve	Before Record Submit (On Update) After Record Submit (On Update)	For the Before Record Submit trigger, only applies to record types such as Sales Orders, Expense Reports, Time Bills, Purchase Orders, and Return Authorizations.
Cancel	Before Record Submit (On Update) After Record Submit (On Update)	—
Copy	Before Record Load (On Update)	—
Create	Before Record Load (On Create) Before Record Submit (On Create) After Record Submit (On Create)	—
Delete	Before Record Submit (On Update)	—

Event Type	Trigger Types (Workflow Initiation)	Limitations
Direct List Edit	Before Record Submit (On Update) After Record Submit (On Update)	**Note:** Only the fields that are updated during the processing of the record display values. Fields that are not updated by XEDIT are blank during the time that the record is processed by the workflow.
Drop Ship	After Record Submit (On Create)	Only applies to Purchase Orders with items specified as **drop ship**.
Edit	Before Record Load (On Update) Before Record Submit (On Update) After Record Submit (On Update)	—
Edit Forecast	Before Record Submit (On Update)	—
Email	Before Record Load (On Update)	—
Mark Complete	Before Record Submit (On Update)	Only applies when users click **Mark Complete** on Task and Phone Call records.
Order Items	After Record Submit (On Create)	Only applies to Purchase Orders with items specified as **drop ship**.
Pack	Before Record Submit (On Update)	Only applies to Item record types, like Item Fulfillment records.
Pay Bills	After Record Submit (On Create)	Only applies to Vendor Payments accessed through the Pay Bill page.
Print	Before Record Load (On Update)	—
Reassign	Before Record Submit (On Update)	Only applies when users click **Grab** on Case records.
Reject	Before Record Submit (On Update)	Only applies to record types such as Sales Orders, Expense Reports, Time Bills, Purchase Orders, and Return Authorizations.
Ship	Before Record Submit (On Update) After Record Submit (On Update)	Only applies to Item record types, like Item Fulfillment records.
Special Order	After Record Submit (On Create)	Only applies to Purchase Orders with items specified as **special order**.
View	Before Record Load (On Update)	—

For sample workflow initiation, action, and transition properties for event types, see Examples of Event Types.

Workflow Templates Reference

Workflow templates create predefined workflows that you can use as a starting point to create customized workflows to define typical or common business processes. Each template includes predefined workflow states, actions, and transitions. You can create a workflow based on a template and then customize it, depending on your business requirements.

For more information about creating a workflow based on a template, see Creating a Workflow from a Workflow Template.

The following tables describes the workflow templates provided by SuiteFlow:

Name	Record Type	For more information, see ...
Journal Entry Basic Approval	Transaction > Journal Entry	Journal Entry Basic Approval Template
Purchase Order Basic Approval	Transaction> Purchase Order	Purchase Order Basic Approval Template
Sales Order Basic Approval	Transaction > Sales Order	Sales Order Basic Approval Template
Lead Nurturing	Lead > Campaign Email	Lead Nurturing Template

Important: For the templates to appear in the Workflow Manager, you must have the proper role to view the record type for the template. In addition, the features associated with the record type must be enabled in your account. For more information, see the topic for the appropriate template type.

Journal Entry Basic Approval Template

The Journal Entry Basic Approval Template creates a workflow for approval of Journal Entry records. After you create a workflow based on this template, you can edit the workflow to customize it according to your business needs. See Creating a Workflow from a Workflow Template.

The Journal Entry Basic Approval workflow implements the following rules:

- The workflow initiates after a user clicks **Save** on a Journal Entry record and before the record is saved to the database.
- The supervisor of a user that creates a journal entry or an administrator can approve the journal entry.
- If a user that creates the journal entry has no supervisor, for example, the user is at the top of the user hierarchy, the user can approve their own journal entry.
- Rejected journal entries can be submitted for approval again.
- The journal entry does not exit the workflow after the journal entry is approved. The record is locked to all users except administrators.

This workflow requires the Require Approvals on Journal Entries preference enabled on the account that creates the Journal Entry record. For more information about this preference, see the help topic Require Approvals on Journal Entries Preference.

Workflow Definition Properties

The following table describes the workflow definition properties for the Journal Entry Basic Approval Template's workflow definition:

Section	Property	Value
Basic Information	Record Type	Transaction
	Sub Types	Journal Entry
	Execute As Admin	checked
	Release Status	Not Initiating
	Keep Instance and History	Only When Testing
	Enable Logging	checked
	Inactive	clear
Event Definition	On Create	checked
	On Update	clear
	Condition	None
	Trigger Type	Before Record Submit

Workflow Fields

The following table describes the workflow fields in the Journal Entry Basic Approval Template's workflow definition:

Workflow Field Name	Description
Approver	Stores the List/Record of the employee that must approve the journal entry.
Created By	Stores the List/Record of the employee that created the journal entry.

States, Actions, and Transitions

The following table describes each state in the Journal Entry Basic Approval Template's workflow definition and the actions and transitions in each state:

State Name	Actions	Transitions
Initiation	■ Set Field Value. Sets the value of the **Created By** workflow field. ■ Set Field Value. Sets the value of the **Approver** workflow field. ■ Set Field Value. Sets the **Approved** field on the journal entry to false.	■ Transition to the **Pending Approval** state.
Pending Approval	■ Add Button. Adds the **Approve** button according to the action condition. ■ Add Button. Adds the **Reject** button according to the action condition. ■ Remove Button. Removes the **Approve** button included on the standard Journal Entry record form. This workflow only requires the **Approve** button added by the Add Button action above.	■ Transition to **Approved** state if the approver clicks **Approve**. ■ Transition to **Rejected** state if the approver clicks **Reject**.

State Name	Actions	Transitions
	▪ Set Field Display Type. Disables the **Approved** field on the record form.	
Approved	▪ Set Field Value. Sets the **Approved** field to true when the record enters the state. ▪ Send Email. Sends an email that the journal entry was approved to the employee record in the **Created By** workflow field, when the record enters the state. ▪ Lock Record. Locks the Journal Entry record if the user accessing the record does not have the Administrator role.	None. The journal entry stays in the **Approved** state of the workflow, locked to all users except administrators.
Rejected	▪ Send Email. Sends an email that the journal entry was rejected to the employee record in the **Created By** workflow field, when the record enters the state. ▪ Add Button. Adds a **Resubmit for Approval** button if the user accessing the record is the employee record in the **Created By** workflow field or if the user is an administrator. ▪ Set Field Display Type. Disables the **Approved** field on the record form. ▪ Remove Button. Removes the **Approve** button included on the standard Journal Entry record form. ▪ Lock Record. Locks the Journal Entry record if the user accessing the record is not the user that created the record. Allows the user that created the record to edit the journal entry before resubmitting for approval.	▪ Transition to the **Initiation** state if the user clicks the **Resubmit for Approval** button.

Purchase Order Basic Approval Template

The Purchase Order Basic Approval Template creates a workflow that allows a user to create and edit a Purchase Order record and send it for approval. After you create a workflow based on this template, you can edit the workflow to customize it according to your business needs. See Creating a Workflow from a Workflow Template.

The Purchase Order Basic Approval workflow implements the following rules:

- The workflow initiates on any creation event on a purchase order.
- The supervisor of a user that creates a purchase order must approve the purchase order. Users with no supervisor cannot send the purchase order for approval.
- After the purchase order is submitted for approval, the purchase order is locked to all users except the supervisor.
- Rejected purchase orders can be submitted for approval again.
- The purchase order does not exit the workflow after the purchase order is approved. The record is locked to all users except administrators.

This workflow requires the following features, preferences, and permissions enabled:

- Purchase Order feature. Go to Setup > Company > Enable Features, and enable **Purchase Orders** under Basic Features on the Transactions subtab. See the help topic Transaction-Related Features.
- Purchase Order Approval Routing. Go to Setup > Accounting > Accounting Preferences, and enable Purchase Orders under Approval Routing on the Approval Routing subtab. See the help topic Approval Routing Accounting Preferences.

- Allow Default Email on Purchase Orders using Suiteflow Approval Routing preference. Go to Setup > Accounting > Accounting Preferences and select **Allow** for the preference under Purchasing on the Order Management subtab. See the help topic Order Management Accounting Preferences.
- Purchase Order Approval permission

Workflow Definition Properties

The following table describes the workflow definition properties for the Purchase Order Basic Approval Template's workflow definition:

Section	Property	Value
Basic Information	Record Type	Transaction
	Sub Types	Purchase Order
	Execute As Admin	checked
	Release Status	Not Initiating
	Keep Instance and History	Only When Testing
	Enable Logging	checked
	Inactive	clear
Event Definition	On Create	checked
	On Update	clear
	Condition	None
	Trigger Type	All

Workflow Fields

The following table describes the workflow fields in the Purchase Order Basic Approval Template's workflow definition:

Workflow Field Name	Description
Created By	Stores the List/Record of the employee that created the purchase order.

States, Actions, and Transitions

The following table describes each state in the Purchase Order Basic Approval Template's workflow definition and the actions and transitions in each state:

State Name	Actions	Transitions
Initiation	▪ Set Field Value. Sets the value of the **Created By** workflow field to the current user.	▪ Transition to the **Pending Approval** state when the user clicks the **Submit for Approval** button on the record form.

State Name	Actions	Transitions
	■ Set Field Value. Sets the value of the **Approval Status** field on the record form to **Pending Approval**. ■ Add Button. Adds the **Submit for Approval** button to the record form if the **Created By** workflow field is equal to the current user when the record form loads. ■ Set Field Display Type. Disables the **Next Approver** field on the record form when the record form loads. ■ Set Field Display Type. Disables the **Approval Status** field on the record form when the record form loads. ■ Return User Error. Returns a user error if the current user that created the purchase order does not have a supervisor on the Employee record. ■ Set Field Value. Sets the value of the **Next Approver** field on the record form to the supervisor of the current user.	
Pending Approval	■ Set Field Value. Sets the value of the **Approval Status** field on the record form to **Pending Approval**. ■ Add Button. Adds the **Approve** button according to the action condition. ■ Add Button. Adds the **Reject** button according to the action condition. ■ Lock Record. Locks the Purchase Order record.	■ Transition to **Approved** state if the approver clicks **Approve**. ■ Transition to **Rejected** state if the approver clicks **Reject**.
Approved	■ Set Field Value. Sets the **Approval Status** field to **Approved** when the record enters the state. ■ Send Email. Sends an email that the purchase order was approved to the employee record in the **Created By** workflow field, when the record enters the state. ■ Lock Record. Locks the Purchase Order record if the user accessing the record does not have the Administrator role.	None. The purchase order stays in the **Approved** state of the workflow, locked to all users except administrators.
Rejected	■ Set Field Value. Sets the **Approval Status** field to **Rejected** when the record enters the state. ■ Send Email. Sends an email that the purchase order was rejected to the employee record in the **Created By** workflow field, when the record enters the state. ■ Add Button. Adds a **Resubmit for Approval** button if the user accessing the record is the employee record in the **Created By** workflow field or if the user is an administrator. ■ Lock Record. Locks the Purchase Order record if the user accessing the record is not the user that created the record. Allows the user that created the record to edit the purchase order before resubmitting for approval.	■ Transition to the **Pending Approval** state if the user clicks the **Resubmit for Approval** button.

Sales Order Basic Approval Template

The Sales Order Basic Approval Template creates a workflow that allows a user to create and automatically approve a sales order if the total is less than a specific amount or start the approval process if the total is over a specific amount. After you create a workflow based on this template, you can edit the workflow to customize it according to your business needs. See Creating a Workflow from a Workflow Template.

The Sales Order Basic Approval workflow implements the following rules:

- The workflow initiates on any creation event on a sales order.
- The sales order is automatically approved if the total amount of the sales order is $300 or less and the customer's overdue balance is $0.
- The sales order is also automatically approved if the user that creates the sales order does not have a supervisor.
- If the sales order is not automatically approved, the approval process begins. An approver can edit, approve, reject, or cancel the sales order.
- If the approver rejects the sales order, the user may cancel the sales order, or edit the sales order and resubmit it for approval.
- The sales order does not exit the workflow after the sales order is approved. The record is locked to all users except administrators.

Workflow Definition Properties

The following table describes the workflow definition properties for the Sales Order Basic Approval Template's workflow definition:

Section	Property	Value
Basic Information	Record Type	Transaction
	Sub Types	Sales Order
	Execute As Admin	checked
	Release Status	Not Initiating
	Keep Instance and History	Only When Testing
	Enable Logging	checked
	Inactive	clear
Event Definition	On Create	checked
	On Update	clear
	Condition	None
	Trigger Type	All

Workflow Fields

The following table describes the workflow fields in the Sales Order Basic Approval Template's workflow definition:

Workflow Field Name	Description
Approver	Stores the List/Record of the employee that must approve the sales order.
Created By	Stores the List/Record of the employee that created the sales order.

States, Actions, and Transitions

The following table describes each state in the Sales Order Basic Approval Template's workflow definition and the actions and transitions in each state:

State Name	Actions	Transitions
Initiation	■ Set Field Value. Sets the value of the **Created By** workflow field. ■ Set Field Value. Sets the value of the **Order Status** field on the record form to **Pending Approval**.	■ Transition to the **Approved** state if the user does not have a supervisor, or the **Total** field is less than $300.00 and the **Overdue Balance** for the customer is $0. ■ Transition to the **Pending Approval** state.
Pending Approval	■ Set Field Value. If the **Approver** workflow field is empty, set it to the current user's supervisor. ■ Set Field Value. Appends the value of the **Memo** field on the record form to the system date and the text "Pending Approval...". ■ Send Email. Sends an email to the supervisor to notify of a pending sales order. ■ Lock Record. Locks the Sales Order record if the current user is not the approver. ■ Add Button. Adds the **Approve** button if the current user is the approver. ■ Add Button. Adds the **Reject** button if the current user is the approver. ■ Remove Button. Removes the standard **Approve** button with an id of APPROVE. ■ Remove Button. Removes the standard **Create Deposit** button with an id of CREATEDEPOSIT. ■ Remove Button. Removes the standard **Manage Revenue Recognition** button with an id of UPDATEREVREC.	■ Transition to **Approved** state if the approver clicks **Approve**. ■ Transition to **Rejected** state if the approver clicks **Reject**. ■ Transition to the **Canceled** state if the **Order Status** field has a value of **Canceled**.
Approved	■ Set Field Value. Sets the **Order Status** field to **Pending Fulfillment** when the record enters the state. ■ Set Field Value. Appends the value of the **Memo** field on the record form to the system date and the text "Approved...". ■ Lock Record. Locks the Sales Order record if the user accessing the record does not have the Administrator role.	None. The sales order stays in the **Approved** state of the workflow, locked to all users except administrators.
Rejected	■ Set Field Value. Appends the value of the **Memo** field on the record form to the system date and the text "Rejected...". ■ Send Email. Sends an email that the sales order was rejected to the employee record in the **Created By** workflow field, when the record enters the state. ■ Lock Record. Locks the Sales Order record if the user accessing the record is the employee record in the **Created By** workflow field or if the user is an administrator. Allows the user that created the record to edit the sales order before resubmitting for approval.	■ Transition to the **Pending Approval** state if the user clicks the **Resubmit for Approval** button. ■ Transition to the **Canceled** state if the **Order Status** field has a value of **Canceled**.

State Name	Actions	Transitions
	■ Add Button. Adds a **Resubmit for Approval** button if the current user is the user that created the record or if the current user is an administrator. ■ Remove Button. Removes the standard **Approve** button with an id of APPROVE. ■ Remove Button. Removes the standard **Create Deposit** button with an id of CREATEDEPOSIT. ■ Remove Button. Removes the standard **Manage Revenue Recognition** button with an id of UPDATEREVREC.	

Lead Nurturing Template

The Lead Nurturing Template creates a workflow for lead nurturing campaigns. After you create a workflow based on this template, you can edit the workflow to customize it according to your business needs. See Creating a Workflow from a Workflow Template.

The template requires you to specify a campaign email address to be used as the return address for emails sent by workflows based on this template. Additionally, if you have the Subscription Categories feature enabled, you must specify a subscription category for the template. Marketing is selected by default.

The Lead Nurturing Template implements the following rules:

- The workflow initiates on any create event on a lead record.
- If the lead record's status is changed to Closed Won during the workflow, the record transitions to Exit after the Webinar Invitation and Email Case Study steps.

Several permissions are required to use this template:

- Lists:
  - Marketing Campaigns
  - Marketing Templates
  - Customers
- Setup:
  - Set Up Campaign Email Addresses

Workflow Definition Properties

The following table describes the workflow definition properties for the Lead Nurturing Template's workflow definition:

Section	Property	Value
Basic Information	Record Type	Customer
	Sub Types	Lead
	Execute As Admin	unchecked
	Release Status	Not Initiating
	Keep Instance and History	Only When Testing
	Enable Logging	checked

Section	Property	Value
	Inactive	clear
Event Definition	On Create	checked
	On View or Update	clear
	Condition	None
	Trigger Type	After Record Submit

Workflow Fields

The following table describes the workflow fields in the Lead Nurturing Template's workflow definition:

Workflow Field Name	Description
Webinar.Response	Stores the lead's response to the webinar invitation email.
CaseStudy.Response	Stores the lead's response to the case study email.

States, Actions, and Transitions

The following table describes each state in the Lead Nurturing Template's workflow definition and the actions and transitions in each state:

State Name	Actions	Transitions
Webinar Invitation	■ Send Campaign Email. Sends an email to the lead record with an invitation to the webinar when the record enters the state. This invitation includes a link to an online registration form. ■ Subscribe To Record. If the lead clicks a link in the email, the campaign email response record is updated which triggers the workflow to transition to the next state.	■ Transitions to **Wait 1 Day** if the lead clicks a link in the webinar invitation. ■ Transitions to **Industry Comparison** if the lead does not click a link in the webinar invitation within 14 days. ■ Transitions to **Exit** if the lead's record is converted to be a customer during the workflow.
Industry Comparison	■ Send Campaign Email. Sends an email to the lead record containing industry comparison information when the record enters the state.	None. The lead exits the workflow at this state.
Wait 1 Day	None. The state includes a 1–day delay before transitioning to **Email Case Study**.	■ Transitions to **Email Case Study** after 1 day.
Email Case Study	■ Send Campaign Email. Sends an email to the lead record with a link to a case study when the record enters the state. ■ Subscribe To Record. If the lead clicks a link in the email, the campaign email response record is updated which triggers the workflow to transition to the next state.	■ Transitions to **Wait 3 Days** if the lead clicks a link in the case study. ■ Transitions to **Free Trial Offer** if the lead does not click a link in the case study in 7 days. ■ Transitions to **Exit** if the lead's record is converted to a customer during the workflow.
Free Trial Offer	■ Send Campaign Email. Sends an email to the lead record with a free trial offer when the record enters the state.	None. The lead exits the workflow at this state.

State Name	Actions	Transitions
Wait 3 Days	None. The state includes a 3–day delay before transitioning to **Customer Testimonial.**	▪ Transitions to **Customer Testimonial** after 3 days.
Customer Testimonial	▪ Send Campaign Email. Sends an email to the lead record with a customer testimonial when the record enters the state.	▪ Transitions to **Exit** to complete the workflow.
Exit	None. The record completes the workflow in this state.	None. The record completes the workflow in this state.

Action Examples

The following table shows the examples of actions that you can use when developing workflows:

Action Type	Example
Add Button	Using Buttons to Execute Transitions Using Buttons for Navigation
Send Email	Executing an Action with a Saved Search Condition
Set Field Value Return User Error	Using Conditional Fields with Actions
Create Record Go To Record Transform Record	Setting Field Values in Action Definitions
Create Record Subscribe To Record	Creating and Subscribing to a Record

Using Buttons to Execute Transitions

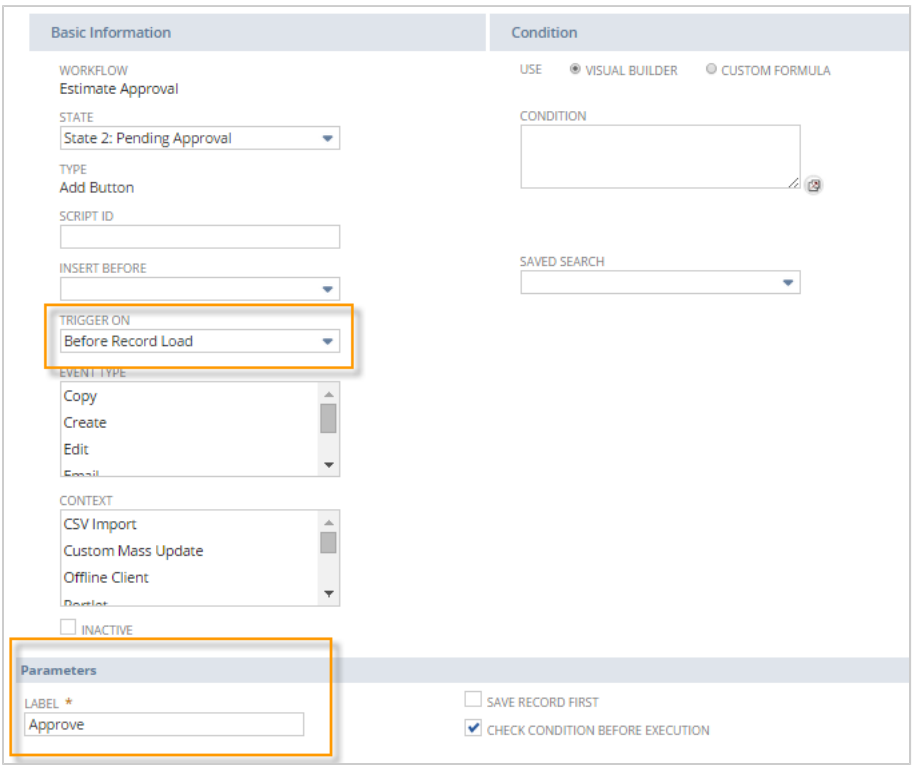
You can use a button to execute a transition in a workflow instead of a server trigger. Use the Add Button action to add a button to a state and create a transition that executes on the button click and moves the record to the next state in the workflow.

For example, add **Approve** and **Reject** buttons to approve or reject a purchase approval. The **Approve** button transitions the record to an Approved state and the **Reject** button transitions the record to a Rejected state.

Use the following example to create an **Approve** button and a transition from **State 2 Pending Approval** to **State 3 Approved**.

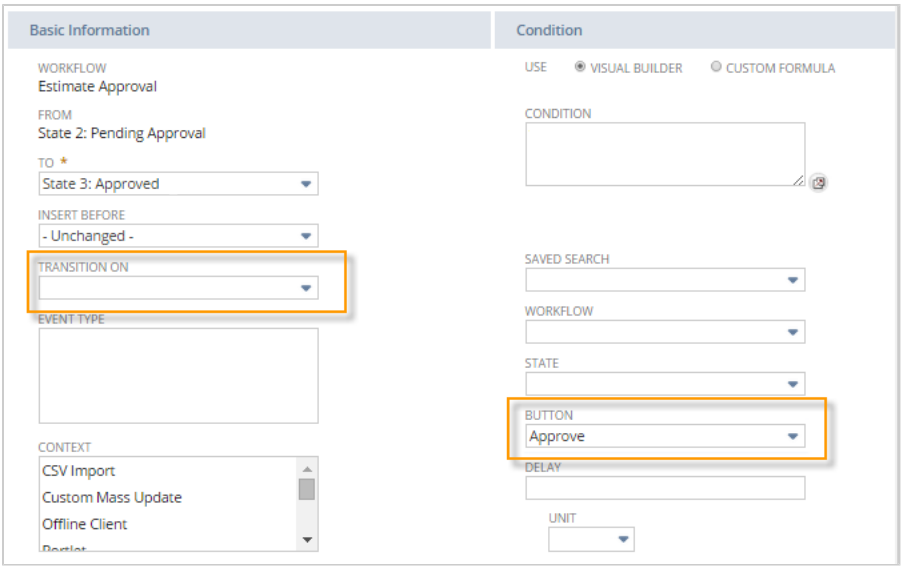
To add a button and use the button to initiate a transition:

1. Open a workflow and select the state that you want to transition from in the diagrammer.
   For example, choose **State 2 Pending Approval**.
2. In the context panel, click **New Action**.
3. In the **New Action** window, click **Add Button**.
4. Make sure **Before Record Load** is selected for **Trigger On** and enter a button label in the **Label** field under **Parameters**. For other Add Button action properties, see Add Button Action Properties.



5. Click **Save**.
6. In the diagrammer, hover over the bottom of the state that you want to transition from. The cursor becomes a filled half-circle.

   Drag the icon to the state to which you want to transition and release the mouse to create the transition.
7. Select the transition in the diagrammer and click the **Edit** icon on the **Transition** tab in the context panel.
8. In the **Workflow Transition** window, leave the **Trigger On** field blank and choose the button you created in the **Button** dropdown list.



9. Click **Save**.

In the above example, when the record enters **State 2: Pending Approval**, an **Approve** button appears on the record form. Clicking the button transitions the record to **State 3: Approved**. Add a Set Field Value action in State 3 to set the approval status of the record to approved.

Note: SuiteFlow prevents workflow instances and actions from processing simultaneously when workflow buttons are clicked multiple times. See, About Multiple Clicked Workflow Buttons.

Using Buttons for Navigation

To use a button for navigation, use a button to transition from a first state to a second state that contains a Go To Page or Go To Record action. The Go To Page or Go To Record action redirects the user from the current record in the workflow to another page, list, or record. The destination depends on the type of action used in the second state.

Use the following example to create a workflow with two states, **State 1** and **State 2**. A transition in **State 1** executes and transitions to **State 2** from a button added with the Add Button action. A Go To Record action in **State 2** redirects the user to a new Customer Deposit record.

To create and use a button for navigation:

1. Create a workflow with the following properties:

Property	Value
Record Type	Transaction
Sub Types	Sales Order
Initiation	Event Based
On Create	checked
On Update	checked
Trigger Type	Before Record Load

For more information, see Creating a Workflow.

Note: Use the Before Record Load trigger because of the Add Button action to add a button in this state. See Before Record Load Trigger and Add Button Action.

2. Save the workflow definition.
3. In the diagrammer, click **New State** twice to create two states: **State 1** and **State 2**.
4. Select **State 1** in the diagrammer, click **New Action** in the context panel, and create an action with the following properties:

Property	Value
New Action window	Add Button
Trigger On	Before Record Load
Label (under **Parameters**)	Create Deposit

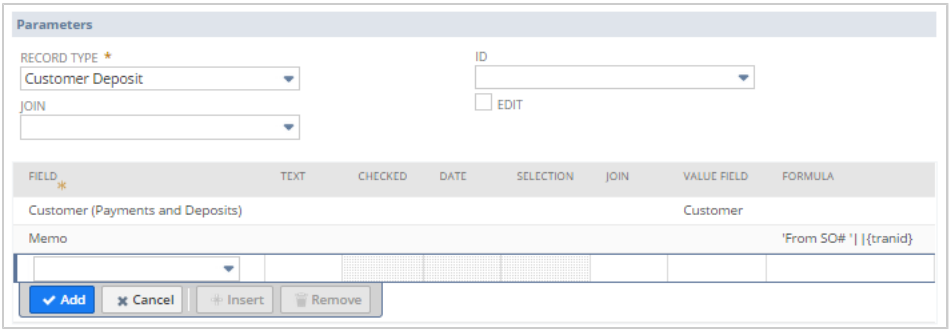
5. Click **Save**.
6. Select **State 2** in the diagrammer, click **New Action** in the context panel, and create an action with the following properties:

Property	Value
New Action window	Go To Record
Trigger On	Entry
Record Type (under **Parameters**)	Customer Deposit

Note: Since the Customer Deposit record does not yet exist, the Go To Record action creates it. See Go To Record Action.

7. On the Go To Record action, under **Parameters**, enter each of the following fields and click **Add**:

Field	Value Field
Customer (Payments and Deposits)	Customer
Memo	'From SO# '||{tranid}



8. Click **Save**.
9. In the diagrammer, hover over the bottom of the **State 1**. The cursor becomes a filled half-circle.

   Drag the icon to **State 2** and release the mouse to create the transition.
10. Select the transition in the diagrammer and click the **Edit** icon on the **Transition** tab in the context panel.
11. In the **Workflow Transition** window, leave the **Trigger On** field blank and choose **Create Deposit** in the **Button** dropdown list. Click **Save**.

To test this workflow, create and save a new sales order. On the sales order, click the **Create Deposit** button to create a deposit record associated with the customer for the sales order. NetSuite creates the deposit record and redirects the user to the new record.

Executing an Action with a Saved Search Condition

You can use a saved search on an action as a condition. The action does not execute unless the record in the workflow is returned by the saved search. When the trigger for the action executes, NetSuite runs the saved search and executes the action if the record is returned in the search. Unlike the behavior for scheduled actions, NetSuite runs the saved search immediately.

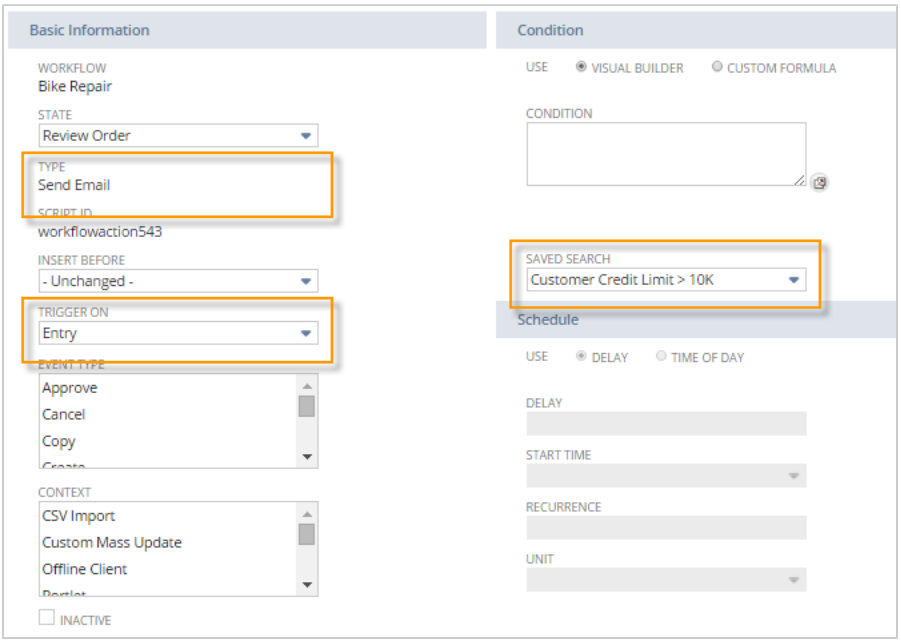
The following example shows how to execute a Send Email action in a state if the customer credit limit is greater than $10,000. You can use a similar method for other actions.

To use a saved search as a condition on an action:

1. Create a saved search for Customer records with a credit limit of more than $10,000. See the help topic Defining a Saved Search.

> **Note:** The record type for the saved search must be the same as the base record type in the workflow definition.

2. Create an action and select the saved search in the **Saved Search** dropdown list for the action. In this example, the saved search is **Customer Credit Limit > 10k**:



In this example, when the record enters the state, the Send Email action executes if the customer associated with the record has a credit limit over $10,000.

> **Important:** Use a server trigger with an action with a saved search as a condition. NetSuite ignores the saved search if the action executes on a client trigger.

Using Conditional Fields with Actions

As a user interacts with a record form, the user makes changes to the fields on the record. NetSuite can detect when a user makes a change to a specific record field. You can direct the workflow to dynamically perform other actions based on when the field changes and the associated field value.

Use client triggers on a record form to detect the events that occur when a user interacts with the form. Use the **Client Fields** multi-select box to detect which field or fields changed during the record form events. For example, you can direct an action to populate the **Sales Rep Phone** after a user edits the **Sales Rep Name** field.

You can use the following action types on the following client triggers:

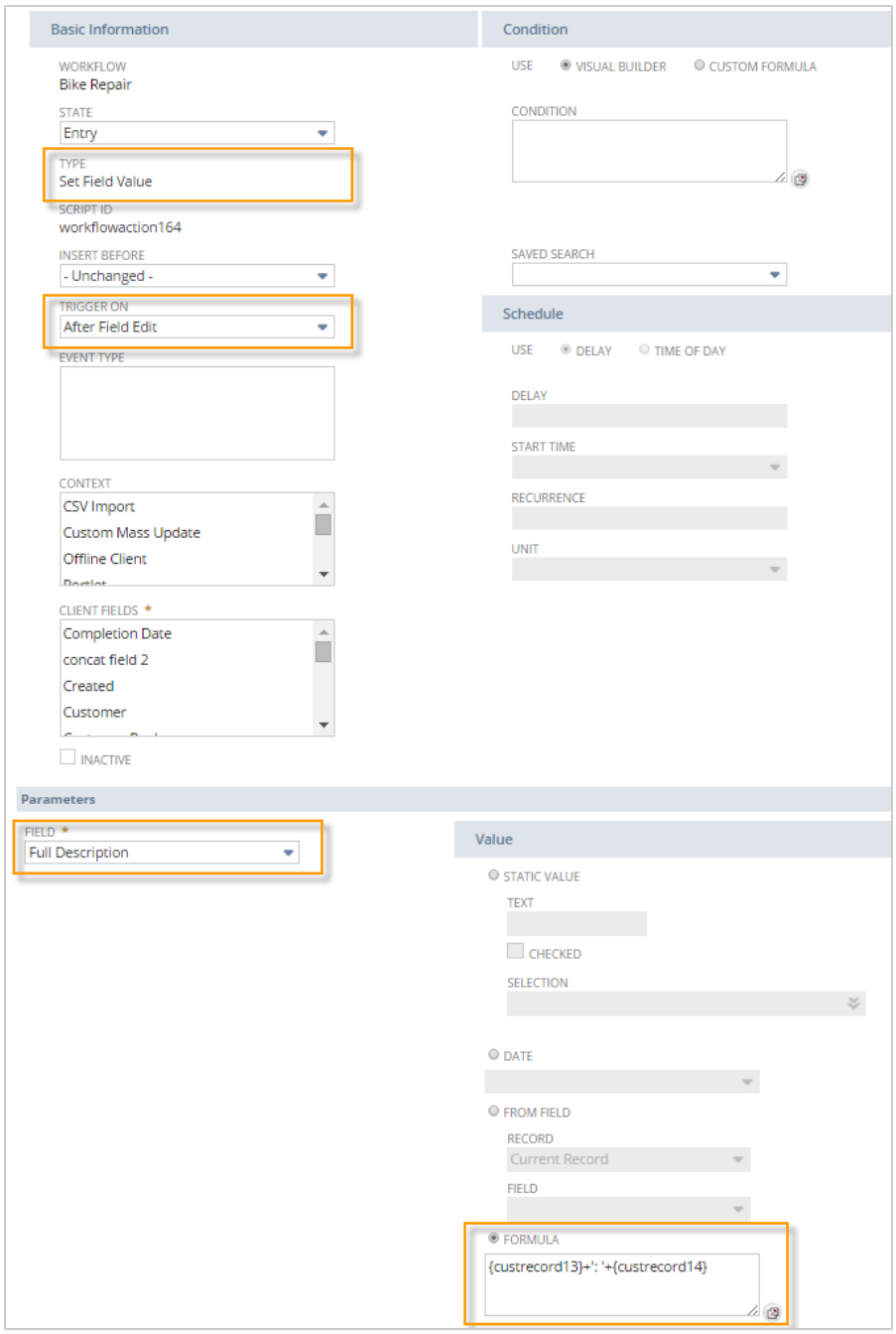
Action Name	Available Triggers
Return User Error Action	Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing, Before User Submit

Action Name	Available Triggers
Set Field Display Label Action	Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing
Set Field Display Type Action	Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing
Set Field Mandatory Action	Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing
Set Field Value Action	Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing, Before User Submit
Show Message Action	Before User Edit, Before Field Edit, After Field Edit, After Field Sourcing, Before User Submit

Dynamic Set Field Value Example

The following example shows a Set Field Value action that executes on the After Field Edit trigger. After a user edits either the **Description** or **Emergency Reason** fields, the action executes. The action concatenates the value of the two fields and populates the **Full Description** field.

You can use a similar solution with client-side actions, like the Set Field Mandatory and Set Field Display Type actions.

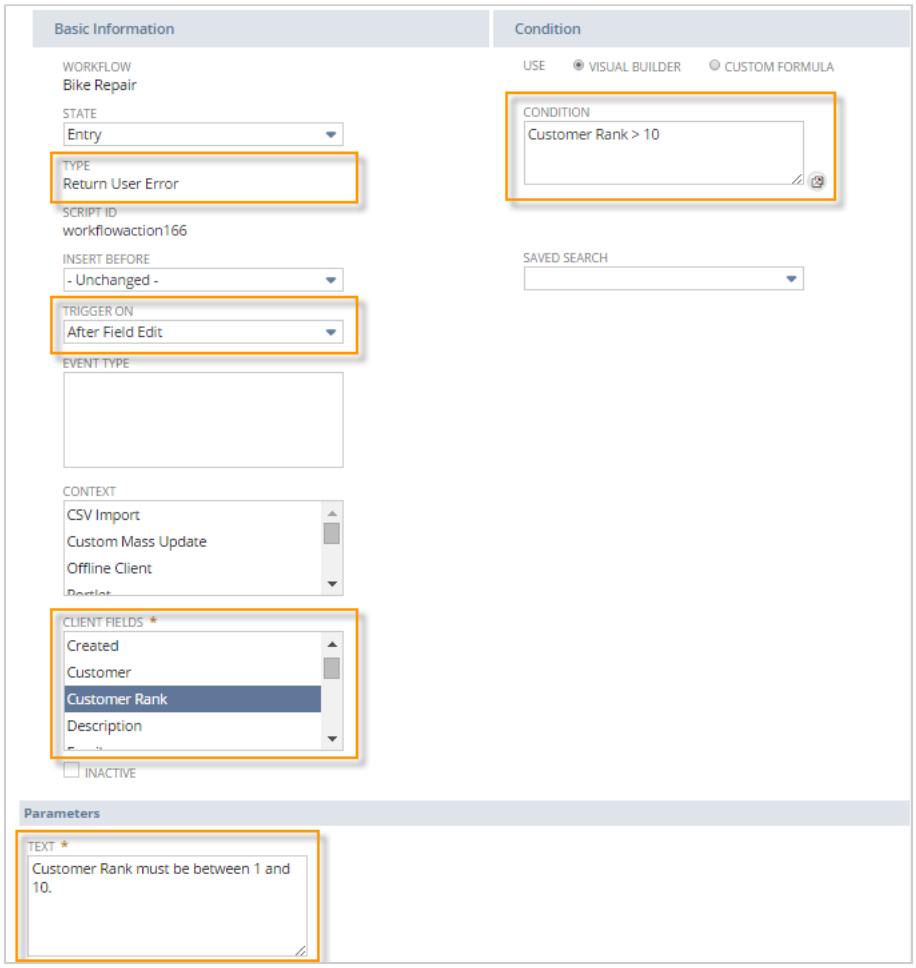


Note: You can reference the internal NetSuite ID of field in formulas. For more information about getting the internal NetSuite ID of fields, see Finding the Internal NetSuite ID.

Dynamic Return User Error Example

The following example shows a Return User Error action that executes on the After Field Edit trigger and uses a condition. After a user edits the **Customer Rank** field, the action executes. The action returns an error if the **Customer Rank** field is greater than 10. NetSuite returns the focus to the **Customer Rank** field and the user cannot proceed further without fixing the rank.

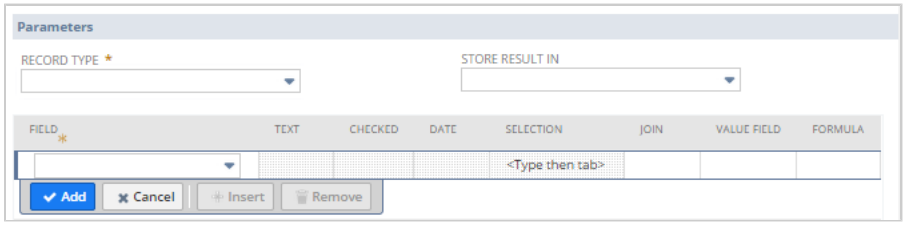
You can use a similar solution with other client-side actions, like the Show Message action, except that Show Message does not prevent the user from continuing. See Return User Error Action, Show Message Action, or After Field Edit Trigger.



Setting Field Values in Action Definitions

When you create a Create Record, Go To Record, or Transform Record action, you can set field values on the target record. You specify the type of record in the **Record Type** dropdown list and then set the field values. You can use the field values from the current record in the workflow or a record that can be joined with the current record. For more information about these action types, see Create Record Action, Go To Record Action, and Transform Record Action.

The following screenshot shows the **Parameters** section for the Create Record action:



The following table lists the columns used by these actions:

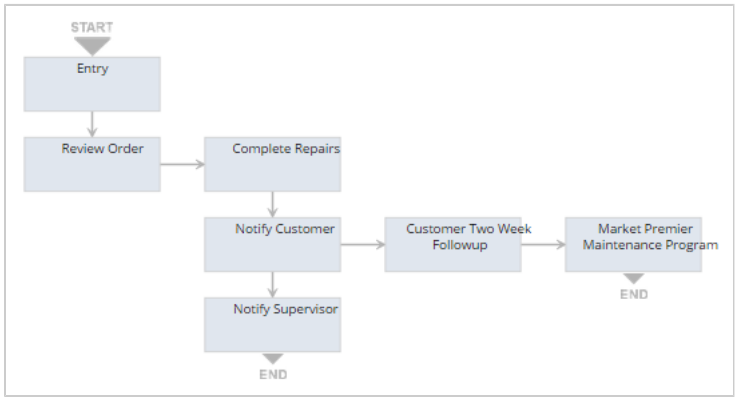
Column	Description
Field	Name of the field for the target record. Select the field name and set the value in the subsequent columns. For more information about the difference between UI field names and field names in target record types, see FAQ: SuiteFlow.
Text	Static text value if the field type for the **Field** column is a text field.
Checked	If the field type for the **Field** column is a checkbox, set the checkbox to checked.
Date	Date value if the field type for the **Field** column is a Date field.
Selection	If the field type for the **Field** column is a List/Record, value for the List/Record. Type and tab to specify a value or click in the box and click the double arrows and select **List**.
Join	List/Record field to join with the current record.
Value Field	Field that contains the value for the field type in the **Join** column.
Formula	Formula that specifies the value for the field type in the **Field** or **Join** columns. Click to open the Formula Builder or enter a value. See the help topic Creating Formula Fields.

Creating and Subscribing to a Record

You can use the Create Record and Subscribe To Record actions to create a record, track the created record, and execute actions and conditions based on the status of the created record. You use a workflow field or state field to reference the created record.

The following example shows how to create a Phone Call record in a workflow for a Bike Repair record, subscribe to the record, and then transition the record in the workflow when the Phone Call record completes.

The following screenshot shows the states and transitions in the workflow:



This example shows how to create and subscribe to a Phone Call record in the **Review Order** state and execute a transition between the **Notify Customer** and **Notify Supervisor** states when the phone call completes.

Use the following basic process to use the Create Record and Subscribe To Record actions together:

1. Create a workflow field. The field is accessible to each state in the workflow. See Step 1 Create the Workflow Field.
2. Use the Create Record action to create a new Phone Call record. See Step 2 Create the Create Record Action.
3. Use the Subscribe To Record action to track the Phone Call record. See Step 3 Create the Subscribe To Record Action.
4. Create a transition that executes when the Phone Call record completes. See Step 4 Create the Conditional Transition.

After you complete the above steps and create the record to initiate the workflow, the Phone Call record is created and the transition executes when the Phone Call record is marked as complete.

Guidelines for Creating and Subscribing to a Record

- You can also use a state field, but all of the actions and any other actions or transitions based on the status of the tracked record must complete in the same state before the record transitions to another state. State fields are accessible from a single state only.
- The Create Record action must appear before the Subscribe To Record action on the **Actions** subtab on the **Workflow State** window.
- You must select a server trigger for both the Create Record and Subscribe To Record action. In general, you should use the same trigger for both actions.

  If you use different triggers for each actions, make sure that the trigger for the Create Record executes before the trigger for the Subscribe To Record action. For example, use the Before Record Submit trigger for the Create Record action and the After Record Submit trigger for the Subscribe To Record action. In addition, make sure both triggers will execute before any transition occurs. For more information, see SuiteFlow Trigger Execution Model.

  This example uses the Exit trigger for both actions, to make sure that they both only execute one time, when the record transitions to a different state.

Step 1 Create the Workflow Field

Use a workflow field to store a reference to the record created with the Create Record action. In this example, the workflow field stores a reference to a Phone Call record. You must specify the type when you create the workflow field. For more information about this example, see Creating and Subscribing to a Record.

To create a workflow field to store a Phone Call record reference:

1. Open a workflow.
2. In the context panel, click the **Workflow** tab, click the **Fields** view, and then click **New Workflow Field**.
3. Enter the following properties:

Property	Value
Label	bikerepair.phonecall
Type	List/Record
List/Record	Call

4. Click **Save.**

Step 2 Create the Create Record Action

Use the Create Record action to create a new record. The record type you create does not need to be the same type as the record in the workflow. In this example, use the Create Record action to create a Phone Call record in the **Review Order** state. For more information about this example, see Creating and Subscribing to a Record.

To create a Phone Call with the Create Record action:

1. If you have not already done so, create the workflow field. See Step 1 Create the Workflow Field.
2. In the diagrammer, select the state, click **New Action** in the context panel, and click **Create Record**.
3. In the **Workflow Action** window, select **Exit** for the **Trigger On** property.

   The action executes when the record leaves the **Review Order** state.
4. In the **Parameters** section, enter the following properties:

Property	Value
Record Type	Phone Call
Store Result In	bikerepair.phonecall (Workflow)

5. In the **Parameters** section, enter each of the following properties and click **Add**:

Field Column	Value Column	Value	Explanation
Title	Text	New bike request to confirm	The title of the Phone Call record.
Assigned	Value Field	Owner	Dynamically set the **Assigned** field of the Phone Call record to the user that creates the Bike Repair record in the workflow.
Message	Text	Please contact the following customer	Tell the assignee of the Phone Call record to phone the customer.
Company	Value Field	Customer	Dynamically set the **Company** field on the Phone Call record to the customer for the Bike Repair record.

6. Click **Save**.

Step 3 Create the Subscribe To Record Action

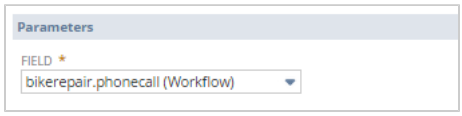
Use the Subscribe To Record action to track a record created in a workflow with the Create Record action. Create the action and choose the instance or state field that stores the reference to the created record. In this example, create a Subscribe To Record action in the **Review Order** state to subscribe to the Phone Call record. For more information about this example, see Creating and Subscribing to a Record.

To create the Subscribe To Record:

1. If you have not already done so, create the Create Record action. See Step 2 Create the Create Record Action.
2. In the diagrammer, select the state, click **New Action** in the context panel, and click **Subscribe To Record**.
3. In the **Workflow Action** window, select **Exit** for the **Trigger On** property.

   The action executes when the record leaves the **Review Order** state.

4. In the **Parameters** section, select the workflow field **bikerepair.phonecall (Workflow)** for the **Field** property:



5. Click **Save**.

Step 4 Create the Conditional Transition

Use a condition on a transition to specify that the transition executes when a record changes status. In this example, create a transition between the **Notify Customer** and **Notify Supervisor** states that executes when the phone call completes. For more information about this example, see Creating and Subscribing to a Record.

To create a conditional transition:

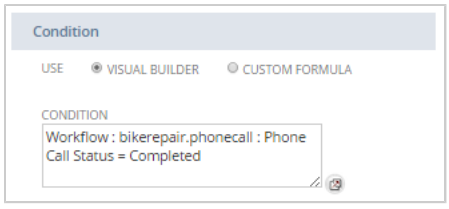
1. If you have not already done so, create the Subscribe To Record action. See Step 3 Create the Subscribe To Record Action.
2. In the diagrammer, hover over the bottom of the state where you want the transition to start. The cursor becomes a filled half-circle.

   Drag the icon to the target state and release the mouse to create the transition.
3. In the diagrammer, select the transition and click the **Edit** icon on the **Transition** tab in the context panel.
4. In the **Workflow Transition** window, in the **Condition** section, open the Condition Builder and create the following condition:

Column	Value
Record	bikerepair.phonecall (Workflow)
Field	Phone Call Status
Compare Type	any of
Selection	Completed

   For more information about creating a condition, see Defining a Condition with the Condition Builder.

   This condition states that when the Phone Call record created by the workflow changes status to Completed, the transition between the two states:



   Note: This type of transition does not require a trigger. NetSuite automatically checks for the status of the Phone Call and executes the transition when the condition evaluates to true.

5. Click **Save**.

The setup for the transition and the related condition is now complete. You can use a similar process to implement the functionality in a workflow for other created record types. To return to the example, see Creating and Subscribing to a Record.

Transitions Examples

The following table lists the provided examples for working with transitions in a workflow:

Transition Type	More Information
Transitioning with a blank trigger type	Blank Transition Trigger
Executing a transition with a saved search condition	Executing a Transition with a Saved Search Condition
Transitioning on a completed state in a child workflow	Specifying States for Child Workflow Transitions

Blank Transition Trigger

The trigger type for the **Trigger On** property of a transition determines when the transition executes.

Leave the transition trigger blank in the following situations:

- You want to use a button to transition to another state. See Using Buttons to Execute Transitions.
- You want a child workflow or child workflow state to complete before the transition executes. See Specifying States for Child Workflow Transitions.
- You want the transition to execute immediately, based on a condition or the results of a saved search.

Blank Transition Trigger Example

You can leave the **Trigger On** property of a transition blank if you want a transition to only execute if specific condition evaluates to true, and not on any specific server trigger.

In the following transition, if the estimated cost of a travel request exceeds the amount that the current approver is allowed, the record transitions from the **Evaluate Approval Limit** state to the **Next Approver** state. With a blank transition trigger, NetSuite evaluates the condition and executes the transition accordingly.

Note: The **Evaluate Approval Limit** state also includes another transition in case the amount of the travel request can be approved by the **Current Approver**.

Executing a Transition with a Saved Search Condition

You can use a saved search as a condition for transition execution. The transition executes if the current record in the workflow is returned by the saved search. Use this type of transition, for example, if you want to transition based on data that is not directly available to the workflow, like data on a sublist.

Note: To appear in the **Saved Search** dropdown list in the transition properties, the saved search record type must be the same as the base record type for the workflow.

To create a transition based on the results of a saved search:

1. Create a saved search on the base record type for the workflow definition. See the help topic Defining a Saved Search.

2. Create a transition between two states in a workflow. See Creating a Transition.

   The behavior of the transition depends on the value of the **Trigger On** property for the transition:

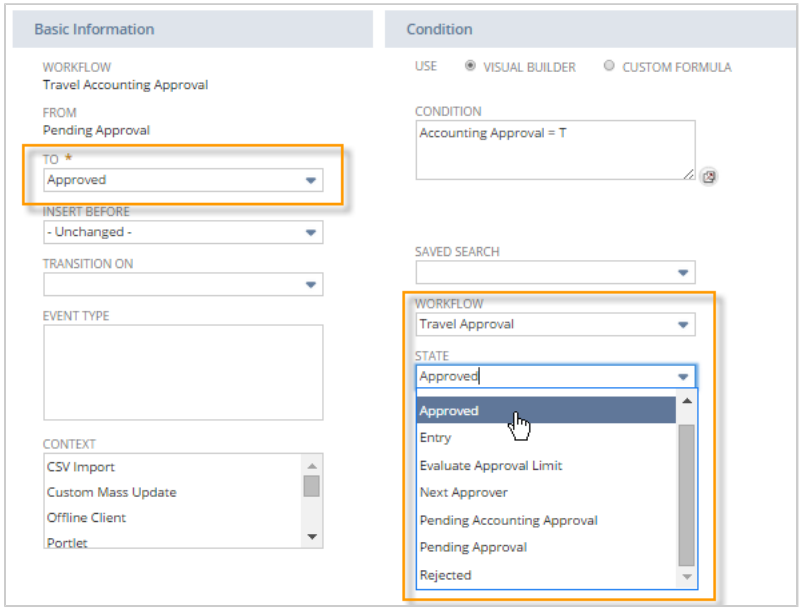
Trigger On Property	Transition Behavior
Blank	Transition executes if the processed record is returned in the saved search results.
Scheduled	Transition executes after the value of the **Delay** property is reached and if the record in the workflow is returned by the saved search.
Entry, Before Record Load, Before Record Submit, After Record Submit	Transition executes on the specific trigger, if the record in the workflow is returned by the saved search.

3. Save the transition.

Specifying States for Child Workflow Transitions

You can use the completion of a child workflow or the completion of a state in a child workflow as a condition for a transition. For example, if a child workflow completes on an **Approved** state, you can direct the child workflow to transition to another state in the parent workflow.

The following screenshot shows the **Workflow Transition** window for a child workflow. The values in the Workflow and State dropdown lists refer to the parent workflow and states in the parent workflow, respectively. When the child workflow leaves the **Approved** state, it transitions to a state in the parent workflow.



Note: If no state in selected in the parent workflow, the child workflow transitions back to the parent workflow on any completion of any end state in the child workflow.

Condition Examples

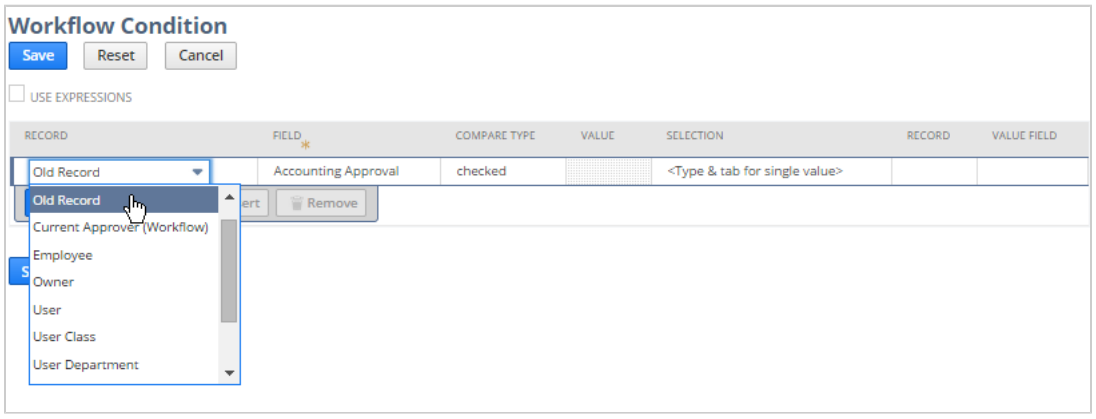
The following table lists the provided examples for working with conditions in a workflow:

Condition Type	More Information
Using the **Old Record** property in the Condition Builder	Referencing Old (Pre-edit) Values in a Workflow
Conditions on a credit hold field for a Customer record	Defining Conditions for Customer Credit Hold Field

Referencing Old (Pre-edit) Values in a Workflow

A workflow can change the values of a record when the workflow runs. When you create conditions for workflow initiation, actions, or transitions, you can reference the values of the record that existed in the database at the point where the user started saving the changed record. The old record is obtained before the first Before Submit workflow is executed and is used for both Before Submit and After Submit workflows.

The Condition Builder in the Workflow Manager contains the **Record** dropdown list. Use this dropdown list to choose other records that can be joined with the current record. You can also use this dropdown list to select the **Old Record** option. Use this value to reference fields on the **Field** dropdown list from the current record in the workflow, before the record entered the workflow.



Important: Use the following guidelines when using pre-edit values for a record.

- Referencing pre-edit values may slow workflow performance because the workflow loads the old record from the database.
- Use a Before Record Submit or After Record Submit trigger when referencing pre-edit values.
- If you want to reference a field of the old record in a condition represented by a formula, the placeholder old needs be used. For example, Old Record:Priority (Task/Call) = High would be represented as {old.priority.id}='HIGH'.

Example of Reacting to a Change in a Field

A common use case for referencing old values in a workflow is reacting to a change in a field on a record. In this situation, you would create a condition similar to Old Record.Some Field != Some Field.

Consider a workflow on a task record that has a Pending Approval state. In this state, you can send an email to the task record assignee when the Priority state is set to High. To do this, you create a Send Email action with a condition for the Priority field.

To create the example Send Email action:

1. Click the Pending Action state in the workflow diagrammer.
2. Click **New Action** in the Context Panel.
3. Click **Send Email**.
4. Set the following properties for the action:
   - **Trigger On**: After Record Submit
   - Use the Condition Builder (**Visual Builder**) to set the following condition for the Priority field: Old Record : Priority (Task/Call) != Priority (Task/Call) And Priority (Task/Call) = High
   - **Sender**:
     - **Specific Sender**: checked
     - **Sender**: Specify a sender
   - **Recipient**:
     - **From Field**: checked
     - **Record (Join Field)**: Current Record
     - **Field**: Assigned
   - **Content**:
     - **Custom**: checked
     - **Subject**: Your task has been escalated!
     - **Body**: Your task '{title}' has been escalated to High priority!
5. Click **Save**.

Note: The formula for the condition built with the condition builder represented by a formula is {old.priority.id} = 'HIGH'.

Defining Conditions for Customer Credit Hold Field

You can define a workflow condition based on whether customers have credit holds placed on their accounts. The **Hold** field on the **Financial** subtab of a Customer record indicates the hold status. The methods to define the condition depend on whether the condition executes on a client or server trigger.

On a Client Trigger

For actions that execute on a client trigger, use the Condition Builder to create the following condition:

```
nlapiLookupField('customer',nvl(nullIfEmpty(nlapiGetFieldValue('entity')),-1),'credithold',false)=='ON'
```

On a Server Trigger

The **Hold** field does not directly exist in the NetSuite database. Instead, it is a combination of two boolean fields: **creditholdoverride** and **manualcredithold**. The following table shows how NetSuite generates the value of the **Hold** field:

Hold Field Value	creditholdoverride Value	manualcredithold Value
Auto	F	F
On	F	T

Hold Field Value	creditholdoverride Value	manualcredithold Value
Off	T	F

For conditions on workflow initiation, actions, and transitions, use the following formula with the mapping from the previous table for the condition. For example, to check that the **Hold** field is **On**:

```
{company.creditholdoverride}='F' and {company.manualcredithold}='T'
```

Context Type Examples

Use the following examples for sample workflow definition settings that initiate a workflow instance or execute actions and transitions for a specific context type.

CSV Import

Use the following sample settings to initiate a workflow and execute actions and transitions for a sales order submitted by CSV import:

Task Type	Description	Settings
Workflow initiation	Initiate workflow instance on new sales orders submitted by CSV import.	**Trigger On:** After Record Submit **On Create:** checked **On Update:** unchecked **Contexts:** CSV Import
Action, Transition	Action or transition executes if sales order submitted by CSV import.	**Trigger On:** After Record Submit **Contexts:** CSV Import

User Event Script

Important: The User Event script context only works with NetSuite back-end user events such as email

Use the following sample settings to initiate a workflow and execute actions and transitions for a sales order created with the following user event script:

```
/**
* @NAPIVersion 2.0
* @NScriptType UserEventScript
*/
define(["N/record"], function (record) {
    function createSalesOrd() {
        var salesOrd = record.create({ type: record.Type.SALES_ORDER, isDynamic: true });
        salesOrd.setValue({ fieldId: 'entity', value: 185 });
        salesOrd.setValue({ fieldId: 'memo', value: 'created through user event' });
        salesOrd.selectNewLine({ sublistId: 'item' });
        salesOrd.setCurrentSublistValue({ sublistId: 'item', fieldId: 'item', value: 42 });
        salesOrd.commitLine({ sublistId: 'item' });
        salesOrd.save();
    }

    return {
```

```
        afterSubmit: createSalesOrd
    };
});
```

Task Type	Description	Settings
Workflow initiation	Initiate a workflow instance on new sales orders created with a user event script.	**Trigger On:** After Record Submit **On Create:** checked **On Update:** unchecked **Context Type:** User Event Script
Action, Transition	Action or transition executes on a sales order created with a user event script.	**Trigger On:** After Record Submit **Context Type:** User Event Script

Custom Mass Update

Use the following sample settings to initiate a workflow and execute actions and transitions for a Customer record updated with the following mass update script.

```
/**
* @NApiVersion 2.0
* @NScriptType MassUpdateScript
*/
define(["N/record"], function (record) {
    function customerMassUpdate(params) {
        record.submitFields({
            type: params.type,
            id: params.id,
            values: {
                comments: 'Specifying Customer Mass Update context'
            }
        });
    }

    return {
        each: customerMassUpdate
    };
});
```

Task Type	Description	Settings
Workflow initiation	Initiate workflow instance on a Customer record updated with a mass update script.	**Trigger On:** After Record Submit **On Create:** checked **On Update:** unchecked **Context:** Custom Mass Update
Action, Transition	Action or transition executes on a Customer record updated with a mass update script.	**Trigger On:** After Record Submit **Context Type:** Custom Mass Update

Examples of Event Types

Use the following examples for sample workflow definition settings that initiate a workflow instance or execute actions and transitions for a specific event type.

Print Event Type

Use the following sample settings to initiate a workflow and execute actions and transitions when a user clicks **Print** in view mode on an existing Customer record.

Task Type	Description	Settings
Workflow initiation	Initiate workflow instance on a Customer record printed in view mode.	**Record Type:** Customer **Trigger On:** Before Record Load **On Create:** unchecked **On Update:** checked **Event Types:** Print
Action, Transition	Action or transition executes on a Customer record printed in view mode.	**Trigger Type:** Before Record Load **Event Types:** Print

Mark Complete Event Type

Use the following sample settings to initiate a workflow and execute actions and transitions when a user clicks **Mark Complete** in view mode on an existing Task record.

Task Type	Description	Settings
Workflow initiation	Initiate workflow instance on a Task record marked complete in view mode.	**Record Type:** Task **Trigger On:** Before Record Submit **On Create:** unchecked **On Update:** checked **Event Types:** Mark Complete
Action, Transition	Action or transition executes on a Task record marked complete in view mode.	**Trigger Type:** Before Record Submit **Event Types:** Print

Drop Ship Event Type

Use the following sample settings to initiate a workflow and execute actions and transitions when a user clicks **Approve** in view mode on an existing Purchase Order record.

Task Type	Description	Settings
Workflow initiation	Initiate workflow instance on a Purchase Order record approved in view mode.	**Record Type:** Transaction, Purchase Order

Task Type	Description	Settings
		Trigger On: After Record Submit **On Create:** checked **On Update:** checked **Event Types:** Drop Ship
Action, Transition	Action or transition executes on a Purchase Order record approved in view mode.	**Trigger Type:** After Record Submit **Event Types:** Drop Ship

Workflow Search Examples

The following table lists the provided examples for working with searches:

Example	More Information
Finding active workflow instances that use buttons for transitions.	Using the Button Filter in a Workflow Instance Search

Using the Button Filter in a Workflow Instance Search

Use the **Button** filter in an advanced workflow instance search to view all records that currently have a button that must be clicked for the record in the workflow to transition to the next state. The filter only shows states that contain a button added with the Add Button action.

Use this type of search to find records that require approval, such as expense reports, budgets, and purchase requests. After you create the search, you can add the results of the search to the Reminders portlet on the dashboard. See Adding Workflow Search Results to the Dashboard.

To use the Button filter in an advanced workflow instance search:

1. Go to Reports > New Search.
2. On the **Search** page, click **Workflow Instance**.
3. On the **Workflow Instance Search** page, check the **Use Advanced Search** box.
4. On the **Criteria** subtab, in the **Filter** dropdown list, select **Button**.
5. In the **Workflow Instance Search** window, do one of the following:
   - To search for buttons that were added with the Add Button action and include the selected states, in the **Button** dropdown list, select **any of**.
   - To search for buttons that were added with the Add Button action and do not include the selected states, in the **Button** dropdown list, select **none of**.
6. In the dropdown list, select the states that you want to include in your search. To select multiple states, press the **Ctrl** key and the states that you want to select.
7. Click **Set**.
8. On the **Criteria** tab, continue to define the standard and summary search filters:

Subtab	Description
Standard	Select results based on the value of individual workflow instance values.

Subtab	Description
	For more information, see the help topic Defining Standard Search Filters.
Summary	Select results based on calculated summary property values for a group of workflow instances; this filter is available only if you defined a summary type for a field on the **Results** subtab. For more information, see the help topic Summary Search Filters.

9. On the **Results** subtab, set the sort values for the search results and the search results fields to display, define summary types for results fields, and select additional search result options.
10. To perform the search and display the results in NetSuite, click **Submit**.
11. (Optional) To view the individual record, in the search results, click **Record**.

Note: You can mimic the click of the button to process all or a subset of the records and transition the records to the next state. See Mass Processing Records in a Workflow.

Testing and Troubleshooting Workflows

After you create a workflow, you must test it to verify that the workflow performs correctly and troubleshoot any issues. During the time that you test a workflow, you can use the workflow activity information generated by NetSuite to view current workflow instances for a record, current and past workflow states, and the actions and transitions performed by a workflow instance. Use this activity information to troubleshoot the behavior of the workflow.

Use the following information when testing and troubleshooting workflows:

- Viewing Workflow Activity
- Testing a Workflow
- Troubleshooting a Workflow
- SuiteFlow Best Practices

Viewing Workflow Activity

When a workflow runs on a record, SuiteFlow records the information about the workflow instance activity for the record. You can use this information when you test and troubleshoot workflows. NetSuite displays the activity on the Active Workflows and Workflow History tabs and in the workflow execution log. You can access the workflow information from any record that the workflow ran against.

The following table describes the workflow activity information generated by NetSuite:

Tab/Log	Description	When to Use
Active Workflows Subtab	Current workflow instances running on a record. After a workflow completes, it no longer appears on this tab.	Use the Active Workflows subtab when you want to see which workflow instances are running on a specific record.
Workflow History Subtab	Displays all states entered by the workflow instance. Includes all current and past state visits. Any state visit stays on this tab for the life of the record.	Use the Workflow History subtab to get more information about each state visited by the workflow, and to view the workflow execution log.
Workflow Execution Log	All actions and transitions that executed for a state on a record. Deleted every 60 days. Appears if you enable logging for the workflow and have the administrator permission on the workflow, or if the state generated errors.	Use the workflow execution log when you test and troubleshoot a workflow, to see which actions and transitions executed or did not execute for a state visit.
Workflow change logs in System Notes	Workflow revisions are logged in System Notes. Workflow revisions increment each time a change is made to a workflow. You can view workflow revision numbers by performing System Notes searches. The results of system notes searches display only the workflow revision number. For more detailed information about the revisions listed in search results, go to the Workflow Definition Page History Subtab of the workflow the change was logged for. For more information about system notes searches, see the help topic Searching System Notes.	Perform a System Notes search to see if changes have been made to a workflow.

Tab/Log	Description	When to Use
	Note: You cannot access System Notes for workflows on the System Notes tab of a workflow record. System Notes can be accessed only from a System Notes search.	

In addition, if errors occur during the scheduled workflow processing, NetSuite sends an email to the owner of the workflow with the error messages in the body of the email. For more information, see Error Handling for Scheduled Workflows.

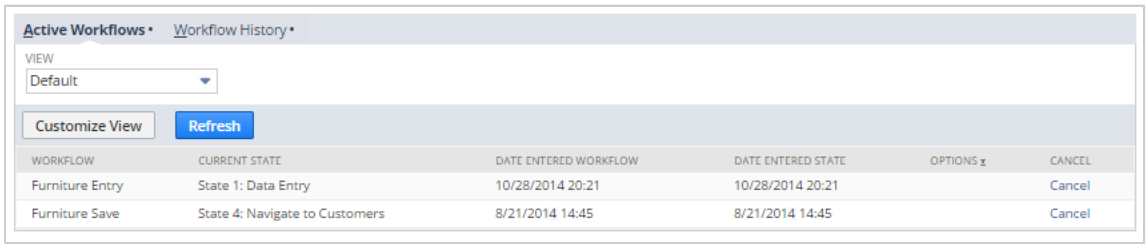
Note: To view workflow activity, users must have the Workflow permission assigned to their roles and SuiteFlow must be enabled in their accounts. For details, see Required Permissions for SuiteFlow.

Active Workflows Subtab

Use the Active Workflows subtab to get more information about the workflow instances that are currently running for a specific record. This subtab only shows active workflow instances for a record. After a workflow completes, it no longer appears on this subtab.

To access the **Active Workflows** subtab, select the record that the workflow runs on, click the System Information tab, and then click the **Active Workflows** subtab. You can also click **Customize View** to add or remove columns on the Active Workflows tab and save the customized view.

The following screenshot shows the **Active Workflows** subtab with an active workflow currently in **State 1 Data Entry**:



The following table describes the default columns displayed on the Active Workflows subtab:

Option	Description
Workflow	Name of the workflow.
Current State	Current state for the workflow.
Date Entered Workflow	Date and time when the workflow initiated.
Date Entered State	Date and time when the workflow entered the current state. Use the Workflow History tab to get more information about actions and transitions that executed for this state, or any other state previously visited by the workflow.
Options	Lists any fields associated with the state that the workflow is currently running. Includes workflow fields and state fields and their current values. For more information about creating and using custom fields, see Workflow Custom Fields.
Cancel	Allows administrators to stop the current workflow instance. The cancel operation only cancels the current workflow instance. Other workflow instances on other records are not stopped.

Option	Description
	Click **Cancel** to stop the workflow.

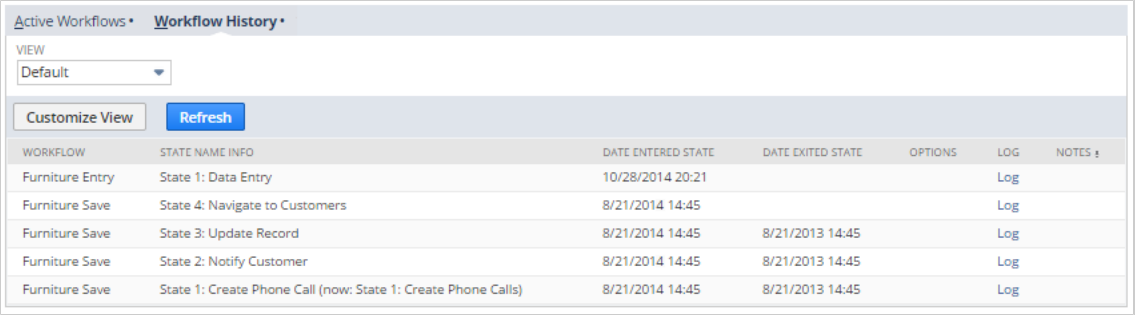
Workflow History Subtab

Use the **Workflow History** subtab to get more information about the states visited by workflow instances on a record. The subtab lists all states entered by the workflow, including any state that the workflow is currently in. A workflow state and the associated workflow execution log appears on the **Workflow History** subtab each time that the workflow enters a state. The state visit stays on the subtab for the life of the record, whereas the workflow execution log is deleted every 60 days.

For example, a workflow has 3 states: State 1, State 2, and State 3. A record enters the workflow in State 1 and then transitions to State 2. From State 2, the workflow transitions back into State 1, and has not entered State 3. Consequently, the states appear in the following order: 1, 2, 1.

To access the **Workflow History** subtab, select the record that the workflow instance ran on, click the **System Information** tab, and then click the **Workflow History** subtab.

The following screenshot shows the **Workflow History** subtab:



The following table describes the default columns displayed on the Workflow History tab:

Option	Description
Workflow	Name of the workflow.
State Name Info	The name of the state in the workflow. This column also indicates any state that was deleted or renamed after the workflow exited the state. For more information, see Deleted and Renamed States.
Date Entered State	Date and time when the record entered the state.
Date Exited State	Date and time when the record exited the state. If the value is empty, the record is currently in the state.
Options	Lists any custom fields associated with the state. Includes workflow fields and state fields. For more information about creating and using custom fields, see Workflow Custom Fields.
Log	Enables you to access the workflow execution log. Click **Log** to open the workflow execution log in a new window. For more information, see Workflow Execution Log.
Notes	If a workflow was canceled, this column shows the user name of the user who canceled the workflow when the workflow was in the state.

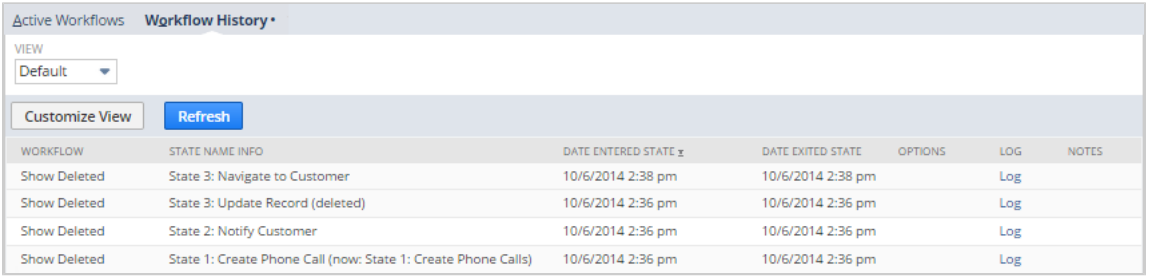
Note that errors are not logged on the History subtab. This is because when an error occurs during a workflow's execution, the workflow is rolled back and the instance of the workflow no longer exists in the database. As such, errors cannot be logged for an instance of a workflow that no longer exists in the database.

Deleted and Renamed States

When you delete or rename a state in Workflow Manager, the workflow history reflects the changes to the state in the **State Name Info** column on the **Workflow History** subtab. This lets you see changes to the states made to the workflow definition directly in the **Workflow History** tab.

For example, a workflow exits a state named **Update Record** and then you delete the state. For any time the workflow has entered and exited the state, the **Workflow History** subtab shows the state name as 'Update Record (deleted)'. Similarly, if a workflow exits a state named **Create Phone Call** and then you rename the state to **Create Phone Calls**, the Workflow History subtab shows the old and new name for the state. If you rename a state when the workflow is currently in the state, NetSuite updates the **State Name Info** but does indicate that the name was changed.

The following screenshot shows deleted and renamed states on the **Workflow History** subtab:



Workflow Execution Log

The workflow execution log is a system-generated log that tracks all actions and transitions that execute on a record for a specific state visit in a workflow. You can use the workflow execution log to debug and troubleshoot the workflow execution for a record. For example, you can use the log to determine which actions and transitions in a workflow did or did not execute.

To access the workflow execution log, select the record that the workflow instance ran on, and click the **System Information** subtab. On the **System Information** subtab, click the **Workflow History** subtab, then click **Log** for the state you want to view. NetSuite displays all log entries for the selected state visit in a new window.

You can use the workflow execution log to view the following types of information:

- Server Triggers
- Event Types and Contexts
- Action and Transitions Status
- Evaluation of Conditions
- Rejected Actions and Transitions

Note: If the overall workflow execution fails, SuiteFlow rolls back any changes made by the actions in the workflow. However, the actions will still appear in the log.

Server Triggers

The workflow execution log displays each server trigger and the associated event type and context type for the trigger. Use this information to view the order in which actions and transitions executed and their corresponding triggers. You can expand and collapse each trigger to view the actions and transitions and the associated status for the action or transition.

The log displays the executed triggers for a specific workflow instance and a specific state visit in that workflow instance. Depending on the workflow definition and the events that occur on the record, for example, viewing and saving the record, the workflow may enter the same state multiple times. Consequently, multiple entries for the same state may appear in the workflow history. The workflow history does not include any state not entered by the workflow.

For an example of a record entering a state multiple times, see Approval Workflow Example.

Note: Client triggers do not appear in the workflow execution log.

The workflow execution log displays the server triggers in the order that the triggers execute. They do not execute in a fixed order. The order in which the triggers appear depends on the workflow definition and the events on the record.

The following table lists the server triggers:

Trigger	Description
ONENTRY	Record enters the state.
ONEXIT	Record exits the state.
BEFORELOAD	Before the record is loaded.
BEFORESUBMIT	Before the record is submitted to the database.
AFTERSUBMIT	After the record is saved to the database.
SCHEDULED	NetSuite scheduler executes the workflow based on the workflow schedule.

Additionally, a server trigger can be record, workflow, or time based:

- Record based. An event occurs on a record. For example, a record is loaded (BEFORELOAD) or saved (BEFORESUBMIT or AFTERSUBMIT).
- Workflow based. An event occurs on the workflow. For example, a workflow enters a state (ONENTRY) or exits a state (ONEXIT).
- Time based. The scheduled time for the workflow arrives (SCHEDULED).

Workflows use the following rules for the order in which the server triggers appear:

- If a workflow moves from state A to state B, the workflow first executes the ONEXIT trigger in state A and then the ONENTRY trigger in state B.
- If a user edits a record, the workflow executes the BEFORELOAD trigger when it loads the record. If a user then saves the record, the workflow executes the BEFORESUBMIT and AFTERSUBMIT triggers, in that order.
- For a specific state visit, only the triggers related to that state visit display.
- The log always starts with the ONENTRY trigger. If the record leaves the state, the last trigger in the log is always ONEXIT. ONEXIT never appears before ONENTRY.
- The BEFORELOAD, BEFORESUBMIT, and AFTERSUBMIT triggers may not appear in a log for a state visit, because the workflow may transition to another state or the workflow may finish.

Event Types and Contexts

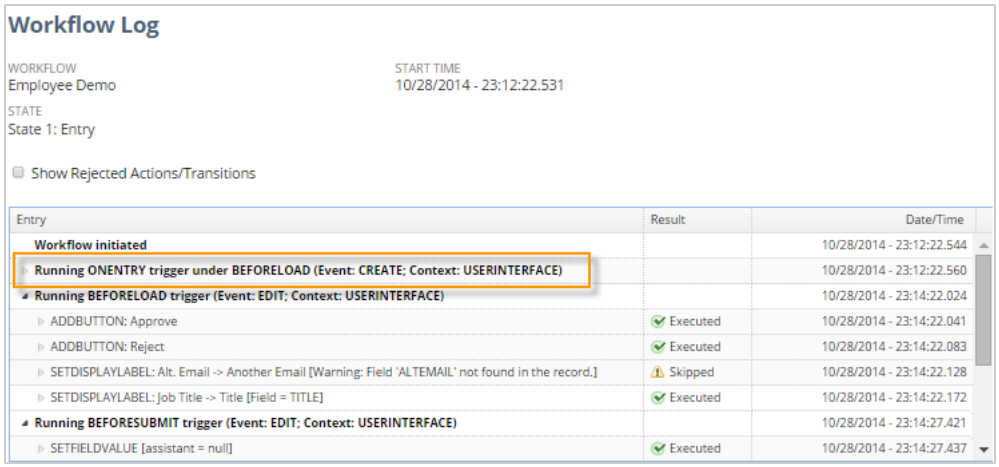
For each server trigger, the log lists the event and context. In addition, the log lists the server trigger on which the workflow entered the state. For example, if the workflow entered the state on a Before Record Load trigger, and the user viewed the record, the ONENTRY trigger displays as:

`Running ONENTRY trigger under BEFORELOAD (Event: CREATE; Context: USERINTERFACE)`

You can optionally set the context and event type for workflow initiation for specific use cases in the workflow definition. However, even if you do not specify the context or event type in the workflow definition, the log displays the context and event for each server trigger. For more information about event types and contexts, see the following:

- Workflow Event Types
- Event Types Reference
- Execution Context Types

The following screenshot shows a workflow execution log for a record that entered a state on a Before Record Load trigger:



Action and Transitions Status

The workflow execution log shows the results of each action or transition.

If an action or transition uses a server trigger that occurs before the trigger on which the workflow enters the state, the action or transition does not execute and does not appear in the log. For example, an Add Button action uses a Before Record Load trigger. If the workflow enters the state on an After Record Submit trigger, the workflow does not execute the Add Button action.

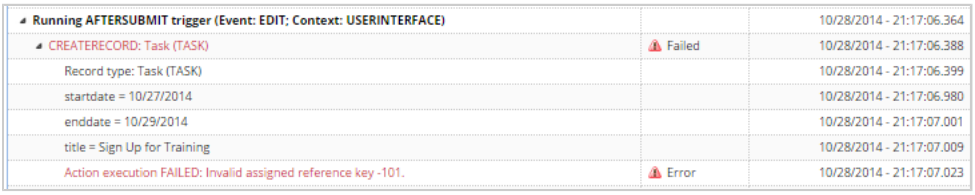
The following table lists the results that may be shown in the log for each action or transition:

Result	Description
Executed	Indicates that the action or transition completed successfully.
Failed	Indicates that the action or transition did not complete. The log lists the reason for failure. The workflow skips subsequent actions for the specific server trigger. If you run the workflow again, the same actions will attempt to execute, and may fail again.
Skipped	One of the following events occurred: ■ A non-critical error occurred to prevent an action or transition from executing, but this error did not stop the execution of subsequent workflow actions or transitions. ■ The action is not supported by the trigger on which the record entered or exited the state. For example, if a record enters a state on a Before Record Load trigger and you set the trigger for a Send Email action as Entry, the Send Email action is skipped. Use the Trigger On field to view supported triggers for the Send Email action. For more information, see Skipped Actions.

Result	Description
	For skipped actions, the log also includes a Warning for any errors associated with the action.
Considered	One of the following occurred: ■ A condition created with the Condition Builder or a custom formula on an action or transition evaluated to false. ■ The action or transition includes a saved search as a condition, and the record was not returned by the saved search. ■ The record was set to transition after another workflow completed or after another workflow exited a state. However, the other workflow had not completed or the other workflow had not finished with the specified state.

In addition, for each action or transition, the workflow execution log includes the properties for the action or transition. You can use the action or transition properties to help troubleshoot the workflow execution.

The following screenshot shows the log for a failed Create Record action:



Evaluation of Conditions

The workflow execution log shows the results of each evaluated condition for actions and transitions. For each condition, the log displays the condition that was evaluated and either True or False for the condition. You can use this information to find out why an action or transition did or did not execute because of a condition.

The log includes evaluation of the following types of conditions:

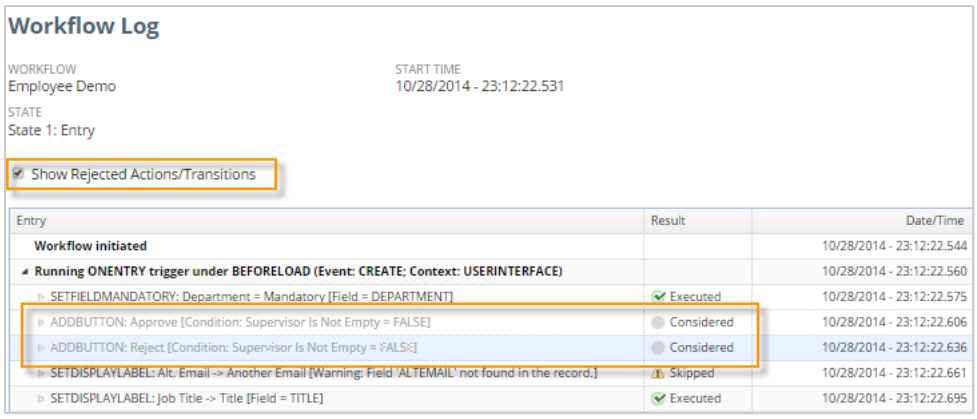
- Condition created with the Condition Builder
- Custom formula
- Saved search
- Completion of another workflow or completion of another workflow within the specified state (transitions only)

The following screenshot shows two conditions that evaluated to false, and consequently, the Return User Error action did not execute:

Rejected Actions and Transitions

If you select the **Show Rejected Actions/Transitions** box, the log displays actions and transitions that were rejected, for example, actions with conditions that evaluated to false. Use this option to see why the actions or transitions were considered and then not executed.

The following screenshot shows an example of considered actions that appear in the log when you select the **Show Rejected Actions/Transitions** box:



Asynchronous Execution for Workflows

The workflow execution log indicates which workflow tasks were processed asynchronously. The workflow tasks that execute asynchronously are placed in the scheduling queue and execute in the order in which they are placed in the queue. You can direct the Workflow Manager to execute workflow tasks asynchronously using the `task.WorkflowTriggerTask` object. For more information, see the help topic task.WorkflowTriggerTask.

For example, for asynchronous execution, the workflow execution log displays the following message:

```
Running ONENTRY trigger under AFTERSUBMIT (Event: CREATE; Context: WEBSTORE) Asynchronous execution
```

Error Handling for Asynchronous Workflow Tasks

When you create an asynchronous workflow task, if an error occurs during the processing of the workflow, NetSuite sends the error details to the owner of the workflow in an email. An error can be a user error or an internal SuiteFlow error. You can use the information in the email message to troubleshoot the workflow.

NetSuite sends an email message for errors that occur with the Asynchronous workflow execution task when initiated using task.WorkflowTriggerTask in SuiteScript 2.x.

Note: NetSuite also sends an email message when an error occurs when a record for a scheduled workflow is loaded or when the record is saved.

The following errors are possible when workflow tasks are processed asynchronously:

- **Errors during workflow execution**. For instance, a workflow creates a new record using the Create Record action, however the action doesn't specify values for all required fields. The Create Record action consequently fails.
- **Skipped record errors**. Errors occurring when two asynchronous tasks are processed in parallel that attempt to process the same record. In such cases, NetSuite prevents records from being processed by two or more asynchronous workflow tasks in parallel. When this happens, it is possible for records to be skipped. For instance, if one task starts processing the record and the second task attempts to as well, the second task must wait until the first finishes. The amount of time the second task waits is limited, and there is a chance the task doesn't have a chance to process the record before the waiting time ends.
- **Infrastructure shutdown.** During maintenance, asynchronous tasks might be stopped. The tasks would finish processing the current record, but would be unable to process any more records until the next scheduled workflow execution. The records that were unable to be processed during the infrastructure downtime are reported as skipped.

In the case of any of the listed errors, NetSuite sends an email to the workflow owner to notify of the errors occurrence.

Error Handling for Scheduled Workflows

When you create an asynchronous workflow task, if an error occurs during the processing of the workflow, NetSuite sends the error details to the owner of the workflow in an email. An error can be a user error or an internal SuiteFlow error. You can use the information in the email message to troubleshoot the workflow.

NetSuite sends an email message for errors that occur with the following types of scheduled workflow tasks:

- Scheduled workflows
- Scheduled actions
- Delayed processing of subscribed records. For example, a workflow instance subscribes to a record using the Subscribe To Record action and the workflow instance triggers when a change occurs on the record. If a large number of subscribed records change, NetSuite may add the workflow instances to the workflow queue and trigger the workflow instances on the next run of the NetSuite scheduler. See Creating and Subscribing to a Record.
- Execute Now feature which manually executes scheduled workflows

Note: NetSuite also sends an email message when an error occurs when a record for a scheduled workflow is loaded or when the record is saved.

The following errors are possible for scheduled workflows:

- **Errors during workflow execution**. For instance, a workflow creates a new record using the Create Record action, however the action doesn't specify values for all required fields. The Create Record action consequently fails.
- **Infrastructure shutdown.** During maintenance, scheduled tasks might be stopped. The tasks would finish processing the current record, but would be unable to process any more records until the next scheduled workflow execution. The records that were unable to be processed during the infrastructure downtime are reported as skipped.

In the case of any of the listed errors, NetSuite sends an email to the workflow owner to notify of the errors occurrence.

NetSuite sends the email with the workflow name in the subject line and the internal record ID and error message in the body of the email. For example, for an error with a workflow named 'Expense Approval' on a record with an ID of 123, the email appears with the following subject and body:

Subject: `Workflow Failure: Expense Approval`

Body: `ID: 123 – End date must be after start date`

The number of emails sent by NetSuite depends on the number of errors that occur. NetSuite sends an email for each of the first five errors in a workflow. If more than five errors occur, NetSuite waits until the workflow processing completes and then sends all errors, including the first five, in a single email message.

Testing a Workflow

Before making a workflow available to all users, test the workflow to make sure that the workflow runs correctly. Complete the following steps to test a workflow:

- Set up the workflow for testing. Enable logging and set the release status. For more information, see Setting Up a Workflow for Testing.
- Test conditions. Set up a testing scenario where you can verify that the conditions can evaluate to both true and false. For more information, see Testing Workflow Conditions.
- Test actions and transitions. Make sure that actions and transitions appear in the workflow execution log and complete successfully. For more information, see Testing Actions and Transitions.
- Test buttons. Make sure buttons appear in all states of the workflow where they are required. For more information, see Testing Buttons.
- Test Send Email actions. Use the workflow execution log to make sure that the Send Email actions send an email. For more information, see Testing a Send Email Action.
- Test scheduled workflows. Initiate a scheduled workflow to start immediately and quickly test the logic in the workflow. For more information, see Testing Scheduled Workflows.
- Test scheduled actions and transitions. Set the smallest possible delay on a scheduled action or transition for testing. For more information, see Testing Scheduled Actions and Transitions.
- Test for user access. Test access for specific roles, test from within the Employee Center, and test access to related records. For more information, see Testing for User Accessibility.

Note: Workflow errors can occur due to incorrect triggers on actions and transitions. Make sure you review triggers and the SuiteFlow trigger execution model. For information, see Workflow Triggers and SuiteFlow Trigger Execution Model.

Setting Up a Workflow for Testing

To set up a workflow for testing, complete the following tasks:

- Set the **Release Status** to **Testing** and make sure you are the owner of the workflow. Only the workflow owner can initiate a workflow in testing mode on a record. However, the workflow actions will execute for any user after the workflow starts. After you test the workflow and verify that it runs correctly, you can set the **Release Status** to released mode. Workflows set to released mode run for any user who has access to the record in the workflow.

  For more information about release statuses and user access to workflows, see Release Status and Workflow Audience.
- Enable logging on the workflow. Enable logging to view the actions and transitions performed by the workflow in the workflow execution log.

  For more information about the **Enable Logging** property, see Creating a Workflow.

Testing Workflow Conditions

If you use conditions in your workflow, use the workflow execution log to make sure that the conditions work as designed. In general, make sure that you create conditions that can be met. Otherwise, the condition can never evaluate to true and the workflow does not run as expected.

Run the following condition tests on a workflow:

- Test a scenario when the condition evaluates to true to make sure that the workflow executes the actions and transitions as expected.
- Test a scenario when the condition evaluates to false to make sure that the workflow does not execute the actions and transitions.
- If you have a condition on the workflow initiation, verify that the workflow does or does not initiate based on the condition.

Tip

If actions or transitions with conditions do not execute, temporarily remove the condition and then run the workflow again. If the workflow runs correctly, the condition has an error.

Testing Actions and Transitions

When you test a workflow, review the workflow execution log to see which actions and transitions did or did not execute during the run of your workflow. Based on log details, you can make changes to the workflow to resolve issues with actions or transitions. For more information about the workflow execution log, see Workflow Execution Log.

Actions and transitions can have one the following statuses:

- Executed
- Failed
- Skipped
- Considered

In addition, an event or transition may not appear in the log because they did not execute. SuiteFlow will only attempt to execute actions of the appropriate trigger type. Make sure that you understand triggers and the Suiteflow triggering model. For more information, see Workflow Triggers.

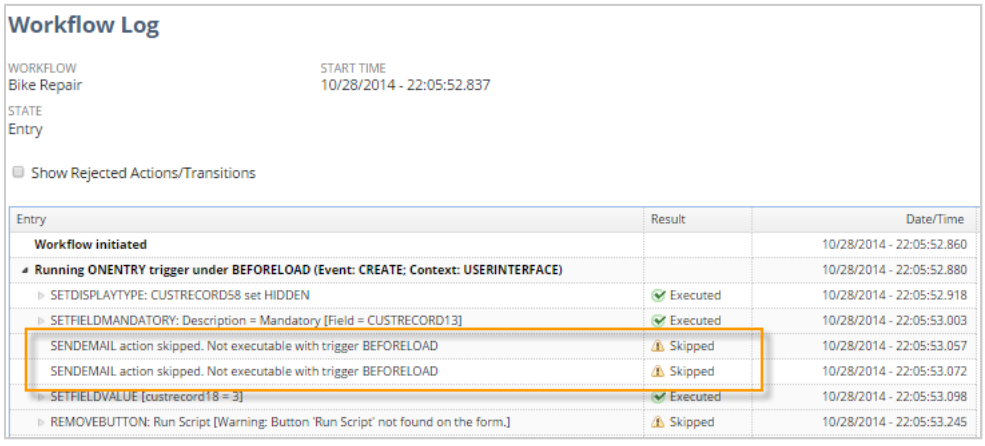
Tip

To test the execution of one specific action within a state, set all other actions in the state to **Inactive**. For each action, select the **Inactive** box on the action definition page and save the action. After you test, you can re-activate the actions. You can also change the trigger type for the action to make sure the action trigger type did not cause the failure.

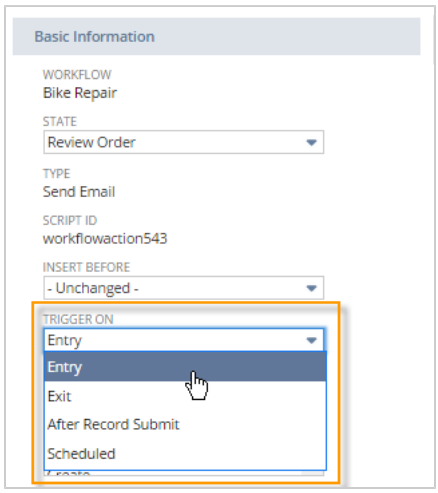
Skipped Actions

When SuiteFlow skips an action, the action may not be supported for the server trigger on which the workflow entered the state.

The following screenshot shows a record that entered a state on a Before Record Load trigger and consequently, the Send Email action was skipped:



You can view the options for the Send Email trigger on the Send Email action definition page in the **Trigger On** dropdown list:



This action can be executed on any of the four triggers listed, depending on the server trigger on which the record enters the state. Since Before Record Load is not an option for the action, the workflow skips the action when running ONENTRY trigger under BEFORELOAD.

Note: The absence of the Before Record Load trigger type for the Send Email action means that even with the trigger type set to Entry or Exit, this action will be skipped if the record enters the state on a Before Record Load or Before Record Submit trigger.

Testing Buttons

You must test Add Button actions in each state where you want the button to appear. For example, if you want a Close Order button to appear on a record form as it moves from state to state, you must add the Add Button action for the Close Order button to every state in the workflow. If you only add it to the first state, the button does not appear on any subsequent state in the workflow.

You can use the workflow execution log to test the context of the record and make sure that the triggers for the Add Button action are set correctly. For example, a workflow initiates on an After Record Submit trigger, indicating the user clicked Save on the record. The workflow does not execute any actions set to trigger before the After Record Submit trigger executes. Since the Add Button actions are set to trigger on Before Record Load, which occurs before the After Record Submit trigger, the workflow does not execute the Add Button actions.

In this example, you can change the workflow initiation trigger to All or Before Record Load. You could also redesign the workflow so that the record enters the state with the Add Button actions on a Before Record Load or Entry trigger.

Note: If you create a workflow that includes an Add Button action, when a user interacts with the button and the record is saved, the System Notes will list the role of the user as Administrator, regardless of the role the user is logged in as. This behavior is expected and is not related to the Execute As Admin property on the Workflow Definition page.

Testing a Send Email Action

You can use the workflow execution log to test if a Send Email action executes. You can use your own email address in the **Cc** fields in both the **Sender** and **Recipient** sections of the Send Email action definition page. Verify that the Send Email action works correctly by viewing the results on the workflow

execution log and checking your email account. If it works as designed, you can reset the correct email address before you set the workflow to a **Released** status.

For more information about the workflow execution log, see Workflow Execution Log.

Testing Scheduled Workflows

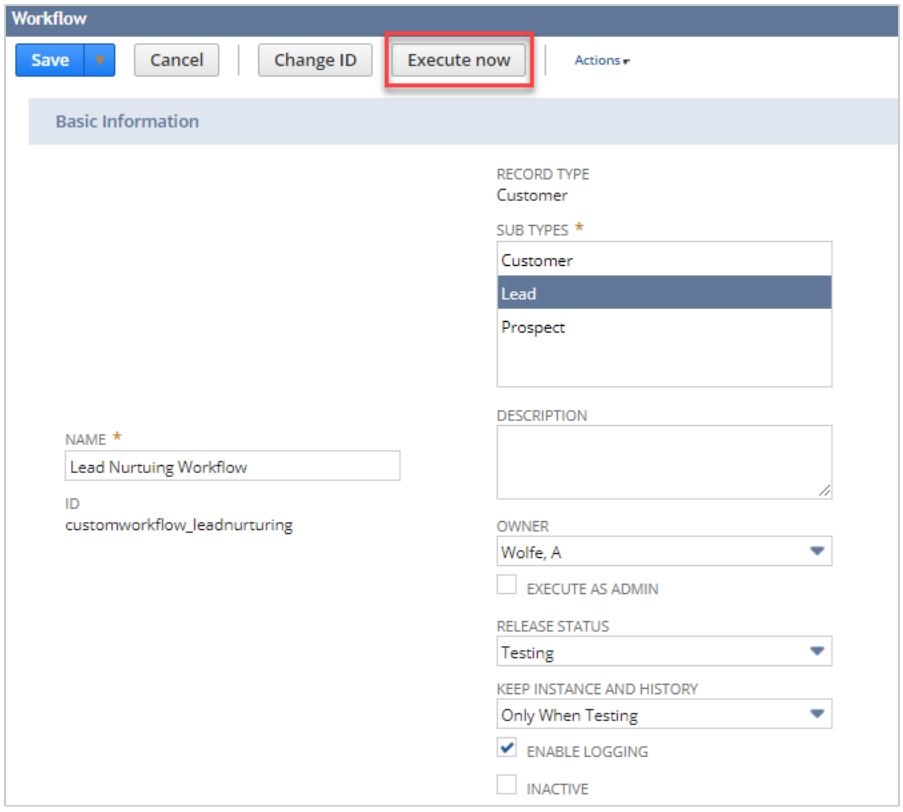
You can execute a scheduled workflow on demand to test a workflow. When you execute a workflow on demand, NetSuite places the workflow in the scheduling queue. The workflow initiates in the order in which it was placed in the queue. The workflow must be set to initiate on a schedule and must be in testing mode. After NetSuite initiates the workflow, you must wait for the workflow to complete before you can execute it again.

Execute a workflow on demand when you want to test the workflow without waiting for the next scheduled run of the workflow. Normally, the NetSuite scheduler evaluates workflows set to initiate on a schedule every 30 minutes. To bypass the NetSuite scheduler, open a scheduled workflow in view mode and click the Execute now button. Executing a workflow on demand immediately tests the logic in a scheduled workflow and reduces the time required for workflow development and testing.

When you execute a workflow on demand, NetSuite only processes the first 20 records returned by the saved search. For example, if the saved search for a scheduled workflow returns 1000 records, the workflow only initiates on the first 20 records returned by the saved search.

The workflow initiates on the first 20 records in any order. For example, a saved search returns records in the order: A, B, and C; the workflow may initiate on record B first, then C and A. In addition, the first 20 records may include the same record multiple times. If the saved search returns repeated records in the first 20 results, each unique record is processed only one time.

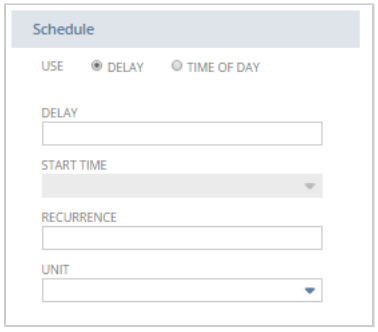
The following screenshot shows the **Execute now** button on the workflow definition page for a scheduled workflow:



Testing Scheduled Actions and Transitions

To test workflows that include scheduled actions or scheduled transitions, set the **Delay** and **Unit** fields to the smallest increments possible. This reduces the amount of time before the scheduled action or transition executes.

The following screenshot shows the **Schedule** options for an action or a transition:



The NetSuite scheduler runs scheduled actions every 30 minutes. Depending on the time you save the settings, the largest amount of time until the next scheduler run is 30 minutes.

Testing for User Accessibility

You must test a workflow for each user role for which you want the workflow to run. When you develop a workflow as an administrator, each workflow runs properly if it is designed correctly. However, you must test the workflow by logging in as the separate user roles for which the workflow will run.

You must test the following accessibility scenarios:

- Access for a specific role. You must test that there are no permission or other restrictions on the record type that would prevent a workflow from running correctly for a specific role. For example, for a lead record that is updated by sales reps, log in with a sales rep role to test the workflow.
- Access from within the Employee Center. If users create or update the record from within the Employee Center, access the record from within the Employee Center with the proper role to test the workflow.
- Access to related records. If a workflow sets a value on a record using the Set Field Value action, and the Set Field Value action gets the value from a related record, you must test that the user role has access to the related record. For example, the record for the workflow is Purchase Order and the workflow runs a Set Field Value action. The Set Field Value action gets the value from a Vendor or Employee record. Verify that any role that works with the Purchase Order has at least View access to the Vendor and Employee records also.

For more information about accessing data in related records, see Defining the Workflow Audience and Execute As Admin.

Troubleshooting a Workflow

Use the workflow execution log and this section to troubleshoot workflow issues for the following areas:

- General workflow issues. Includes issues with a workflow not starting or completing, records hanging on load or not exiting a workflow, and workflow execution limit errors. For more information, see General Workflow Issues.
- Issues with state transitions in a workflow. Includes how to manually transition to a different state in a workflow.

- Actions. Includes issues with actions not executing. For more information, see Action Issues.
- User accessibility. Includes issues with workflows not executing for certain users and limitations on the roles for which a workflow executes. For more information, see User Accessibility Issues.
- Buttons. Includes issues with buttons disappearing and issues removing buttons. For more information, see Button Issues.

For examples on how to troubleshoot a workflow, see Troubleshooting Workflows Examples.

General Workflow Issues

The following table lists common workflow issues and their associated solutions:

Issue	Solution
The workflow did not execute for you during testing.	Take one of the following actions: 1. Make sure that the **Owner** field on the workflow definition page is your user account. If the workflow release status is set to **Testing**, the workflow only runs for the owner of the workflow. 2. If there is a workflow initiation condition, make sure that the record meets the condition and that the condition can evaluate to true. A workflow does not initiate if the workflow initiation condition does not evaluate to true. 3. If the workflow runs on a schedule, make sure that the workflow release status is **Released**. If the release status is **Testing**, the workflow will not run on a schedule.
Users report that when the record in the workflow loads into the browser, the browser hangs and they cannot enter data.	If you set a formula in the Condition Builder, check the syntax of the JavaScript or SQL formula. If you set a workflow action to execute on a client trigger, the formula in the Condition Builder must be a JavaScript formula. If you set the action to occur on a server trigger, the formula in the Condition Builder must be a SQL formula.
You receive the following error message: `Notice (SuiteScript) : Workflow Execution Usage Limit Exceeded`	Take one of the following actions: 1. Check the actions in each workflow state. Make sure that the number of actions in each state do not exceed the 1000 unit usage limit. For more information about workflow governance limits, see Custom Actions and Workflow Governance. 2. Check for an infinite loop. For example, if you have a workflow that transitions from State 1 to State 2, and then from State 2 back to State 2 more than 50 consecutive times, the system identifies it as an infinite loop and a throws the "Workflow Execution Usage Limit Exceeded" error.
In an approval routing workflow, the **Next Approver** field is not set correctly.	For more information, see FAQ: SuiteFlow.
The workflow did not complete its execution.	Take one of the following actions: 1. Determine if the workflow executed. For example, if the workflow runs on an Opportunity record, go to a new Opportunity record to see if the workflow executed on that record. Check the **Workflow History** subtab on the opportunity record to see which states the record entered in the workflow. 2. Check the transitions. For example, an Opportunity record enters a workflow with two states, State 1 and State 2. If the

Issue	Solution
	record did not enter State 2, view the workflow definition and check the transition from State 1 to State 2. Look for errors in the transition definition or any errors in the condition.
Field value in actions or conditions always returns a null value.	If you create a workflow that includes a field value in actions or conditions and it always returns a null value, you could be experiencing an issue with permissions. To resolve this issue, check the **Execute As Admin** box in the workflow definition. For more information, see the help topic Execute As Admin. If you create a workflow that is triggered by the inline edit action, the field value in actions or conditions may return a null value. The reason for this is that the record that the workflow runs on may not be loading completely because of the inline edit action. For more information, see Inline Editing in Workflows.
In a workflow with an Add Button action, the role of the user who interacted with the button is incorrectly listed as Administrator in the System Notes.	If you create a workflow that includes an Add Button action, when a user interacts with the button and the record is saved, the System Notes will list the role of the user as Administrator, regardless of the role the user is logged in as. This behavior is expected and is not related to the Execute As Admin property.

Action Issues

The following table lists common workflow action issues and their associated solutions:

Issue	Solution
You added or edited workflow actions, but the actions do not execute.	Take one of the following actions: 1. Refresh your browser. If you add an action with a server trigger, but the action does not execute when you test the workflow, refresh your browser. For example, you add a Set Field Mandatory action to a state, but the action does not appear to execute. 2. Make sure the trigger type for the action is compatible with the trigger on which the record enters the state. This error can occur when you have a state that includes an action set to trigger on Entry or Exit but the server trigger is not supported by the action. For example, Add Button actions do not run when the record enters the state on a Before Record Submit trigger. For more information, see SuiteFlow Trigger Execution Model. For more information about triggers on actions, see Action Triggers

User Accessibility Issues

The following table lists common user accessibility issues and their associated solutions:

Issue	Solution
The workflow did not execute for other NetSuite users.	Take one of the following actions: 1. Check the **Release Status** field for the workflow. Make sure that the **Release Status** for the workflow is set to **Released**. Workflows set to **Released** will run for any user who has access to the record in the workflow. 2. Check the required fields for actions. For example, if the workflow contains a Create Record action, make sure you set all required fields for

Issue	Solution
	the new record that you want to create. Otherwise, the workflow fails for that user.
Only certain users or certain roles seem to be able to execute a specific workflow.	On the workflow definition page, select the **Execute As Admin** box. Use the **Execute as Admin** property if you want the workflow to execute using administrative privileges, regardless of the permissions of the currently logged-in user. For more information about what users and roles have access to workflows, see Workflow Audience.

Button Issues

The following table lists common Add Button action issues and their associated solutions:

Issue	Solution
Buttons added to a record for the workflow disappear when the browser is refreshed.	Make sure that the Add Button actions trigger on Before Record Load, not on Entry. For more information, see the Workflow FAQ Q: .
Buttons removed from the record for the workflow become visible when the browser is refreshed.	Make sure that the Add Button actions trigger on Before Record Load, not on Entry. For more information, see the Workflow FAQ Q: .

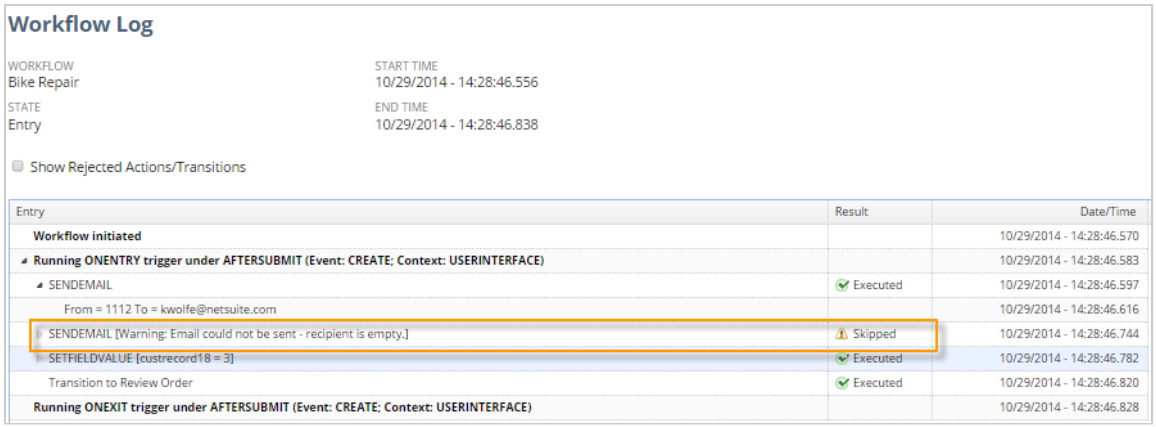
Troubleshooting Workflows Examples

Use the following examples for more details about how to use the workflow execution log to view information and troubleshoot a workflow:

- Actions did not execute. Use this example to get more information about how to troubleshoot a workflow when certain actions did not execute. See Workflow Actions Did Not Execute Example.
- Viewing an approval workflow. Use the workflow execution log to verify that a workflow initiated and that the actions for an approval workflow execute. See Approval Workflow Example.

Workflow Actions Did Not Execute Example

The following screenshot shows the workflow execution log for a Bike Repair workflow that executed on a custom record and entered a state called Entry on an After Record Submit trigger:



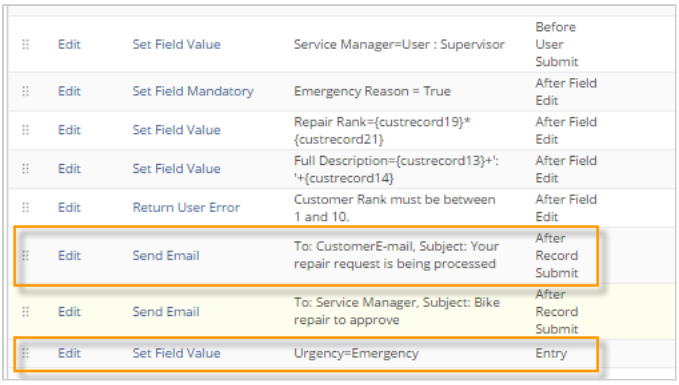
The execution log shows the following information about the workflow:

- The workflow initiated on this record.
- The entry trigger, ONENTRY, ran. In addition, the initiation trigger for the workflow is After Record Submit: `Running ONENTRY trigger under AFTERSUBMIT`.
- A Send Email action executed but the workflow skipped the other Send Email action.
- A Set Field Value action executed.
- The Exit trigger, ONEXIT, executed.

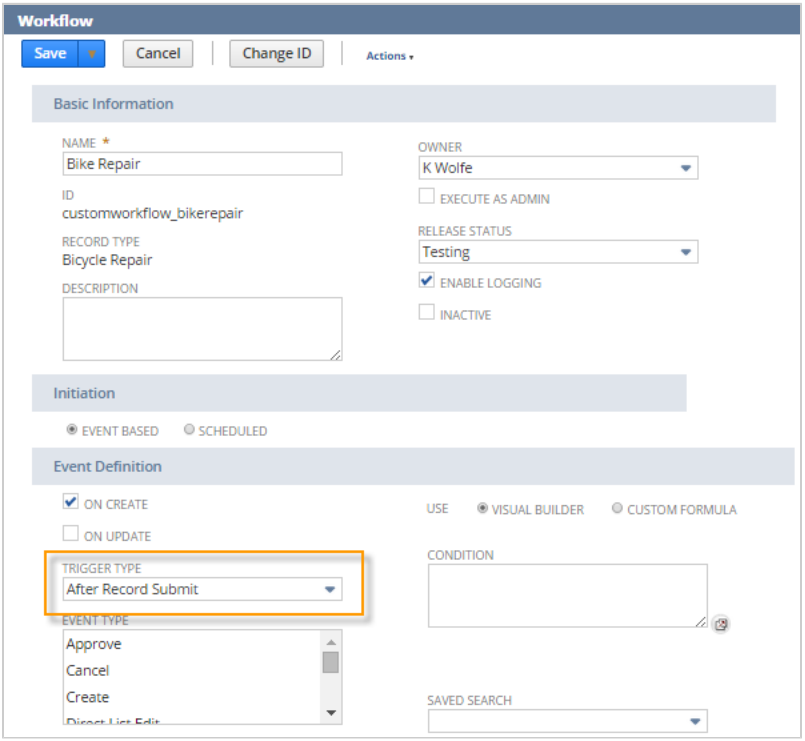
Issues with the Workflow

Although two actions did execute, the workflow state contains many more actions. The actions did not execute because of the server trigger on which the record entered the state.

The following screenshot shows a portion of the **Workflow State** window and the two actions that executed:



The following screenshot shows the initiation trigger type for the workflow:

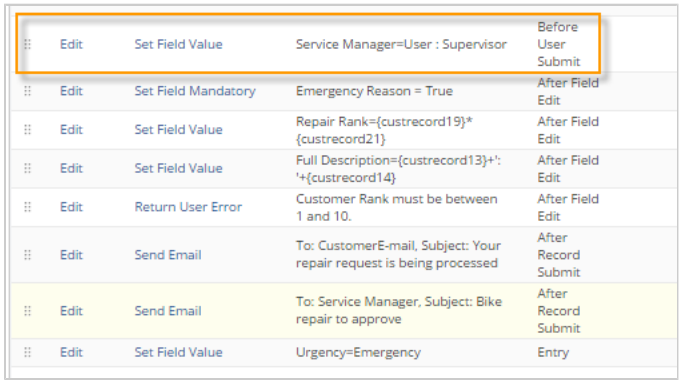


Because the workflow initiated on an After Record Submit trigger, the record entered the state on an After Record Submit trigger, and the following issues occurred:

- The Send Email and Set Field Value actions executed because they were valid actions for a state with an After Record Submit trigger.
- The other Send Email action failed, even though it was set to execute on an After Record Submit trigger, because it required the Service Manager field to be set. However, the Service Manager field was not set because the action that set the value did not execute because of its trigger: Before Record Submit.

This workflow does not initiate until after the data on the record is written to the database. Consequently, any action set to execute before the data is written to the database does not execute. In this example, only actions with an After Record Submit, Entry, or Exit trigger execute.

The following screenshot shows the original configuration of the Set Field Value action:



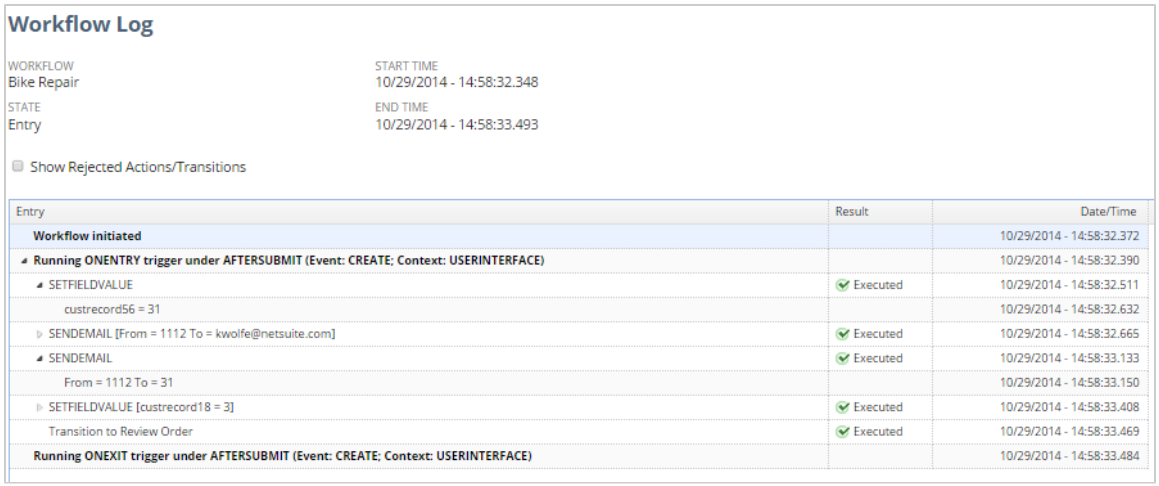
Edit	Set Field Value	Service Manager=User : Supervisor	Before User Submit
Edit	Set Field Mandatory	Emergency Reason = True	After Field Edit
Edit	Set Field Value	Repair Rank={custrecord19}*{custrecord21}	After Field Edit
Edit	Set Field Value	Full Description={custrecord13}+': '+{custrecord14}	After Field Edit
Edit	Return User Error	Customer Rank must be between 1 and 10.	After Field Edit
Edit	Send Email	To: CustomerE-mail, Subject: Your repair request is being processed	After Record Submit
Edit	Send Email	To: Service Manager, Subject: Bike repair to approve	After Record Submit
Edit	Set Field Value	Urgency=Emergency	Entry

Solution

To make sure that the Send Email action to the Service Manager completes, complete one of the following two tasks:

- Change the workflow initiation trigger type to All or Before Record Submit. Consequently, any of the Before Record Submit actions will execute.
- Change the trigger type of the Set Field Value action to After Record Submit. Since the workflow initiates on an After Record Submit trigger, the Set Field Value action will execute.

The following screenshot shows the workflow execution log after the trigger type for Set Field Value action is changed to After Record Submit:



Workflow Log

WORKFLOW
Bike Repair

START TIME
10/29/2014 - 14:58:32.348

STATE
Entry

END TIME
10/29/2014 - 14:58:33.493

Show Rejected Actions/Transitions

Entry	Result	Date/Time
Workflow initiated		10/29/2014 - 14:58:32.372
Running ONENTRY trigger under AFTERSUBMIT (Event: CREATE; Context: USERINTERFACE)		10/29/2014 - 14:58:32.390
SETFIELDVALUE	Executed	10/29/2014 - 14:58:32.511
custrecord56 = 31		10/29/2014 - 14:58:32.632
SENDEMAIL [From = 1112 To = kwolfe@netsuite.com]	Executed	10/29/2014 - 14:58:32.665
SENDEMAIL	Executed	10/29/2014 - 14:58:33.133
From = 1112 To = 31		10/29/2014 - 14:58:33.150
SETFIELDVALUE [custrecord18 = 3]	Executed	10/29/2014 - 14:58:33.408
Transition to Review Order	Executed	10/29/2014 - 14:58:33.469
Running ONEXIT trigger under AFTERSUBMIT (Event: CREATE; Context: USERINTERFACE)		10/29/2014 - 14:58:33.484

After changing the trigger type for Set Field Value action to After Record Submit, the workflow execution log shows that the Set Field Value and Send Email actions, which did not execute before, execute properly.

Workflow Fails Randomly

If your workflow fails randomly, review the following information and try the suggested resolutions.

Issue Description

A workflow executes as expected on multiple transactions or records, but then the workflow randomly does not execute properly. The rate that the workflow fails to execute at is extremely low: the workflow fails to execute properly on approximately one out of every thousand transactions or records. When you check the workflow history for the transaction or record, it shows that the workflow and actions executed but there is no corresponding effect on the transaction or record.

Cause

The workflow fails to execute properly because the Workflow Instance Data and Record Data are not in sync. This behavior is referred to as a Transaction Management problem. The Workflow Instance Data and Record Data are initially synchronized, but when any error occurs, the transaction or record is rolled back by the Transaction Management process while the workflow data is not rolled back. The workflow remains in its current state, which is not the same state that the transaction or record is in. Consequently, the Workflow Instance Data and Record Data become out of sync.

For example, if this error occurred with an approval workflow, the workflow would remain in the approved state and the record would be in a state pending approval.

Resolution

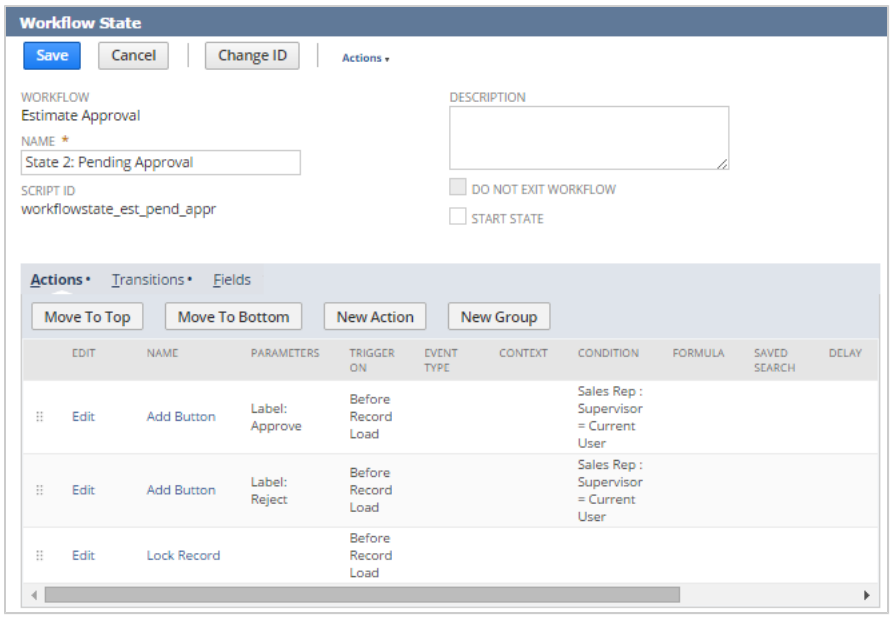
If you have transactions or records that appear to be stuck in a workflow state, try any of the following to resolve the issue:

- Use the Mass Updates feature to move the workflow to the same state that the transaction or record is in. For more information about transitioning an active workflow on a record outside of the defined workflow execution path, see Mass Transitioning Workflow Instances.
- Cancel the current instance of the workflow. Rerun the workflow on the transaction or record.
- Recreate the transaction or record.

Approval Workflow Example

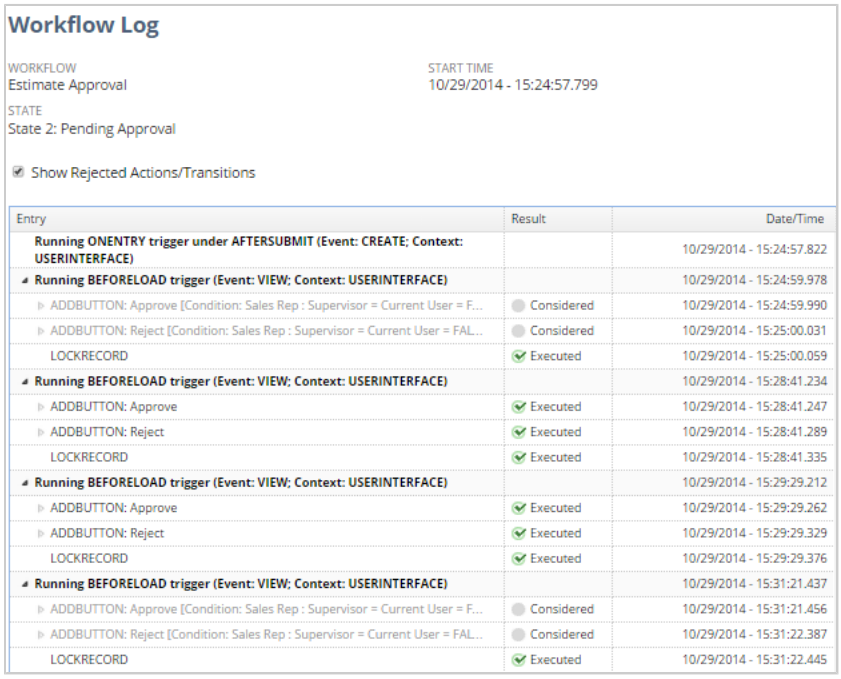
If you create a workflow for a base record type of Estimate, you can view a newly created estimate to see if the record entered the workflow. You can use the workflow execution log to get more details on the specific actions for each state.

The following screenshot shows the configuration of the **State 2 Pending Approval** state in the workflow:



The configuration of the state shows that the **Approve** and **Reject** buttons are only added if the Supervisor for the sales rep views the Quote record. The buttons get added before the Quote record loads into the browser. In addition, the Lock Record action executes on the Before Record Load trigger, regardless of the user that views the record.

The following screenshot shows the workflow execution log for the **State 2 Pending Approval** state:



The workflow execution shows that the user created the quote and the record transitioned to State 2: Pending Approval on an After Record Submit trigger. The subsequent Before Record Load triggers show each time the record is accessed. The first view is by the user, so the Add Button actions do not execute. Only when the supervisor accesses the record do the Add Button actions execute. The log shows that the supervisor viewed the record twice, then the user accessed it, because the buttons were not added.

SuiteFlow Best Practices

Consider applying the following best practices to optimize your workflow performance:

- Combine multiple workflows that have the same logic and functions into a single workflow.
- Filter workflow execution by execution context and event type.
- Convert scheduled workflows that operate on records with static data or data that does not change often to event-based workflows instead. This way, the workflows do not execute on the records unnecessarily.
- Consider the most appropriate workflow execution context for your workflow. The workflow execution context specifies how and when a workflow is triggered to execute. If you don't use the appropriate context, workflows run when they shouldn't and result in unexpected errors. For more information, see the help topic Execution Contexts.

Ordering Workflows

Workflows generally execute in the order that they were created in, but you should not rely on this order when you want to synchronize workflows.

Consider applying the following best practices when you need workflows to execute in a specific order:

- Create a primary workflow that runs other workflows. For example, if you want workflow A to run before workflow B, you create workflow C and set it to initiate workflow A. When workflow A is finished executing, you can set workflow C to initiate workflow B.
- You can synchronize workflows by creating a workflow and configuring it to initiate another workflow as the last action. For example, you can create workflow A and set it to initiate workflow B as the last action in workflow A.

  Important: If you are initiating a workflow from a different workflow, you need to set the release status of the workflow being initiated to **Not Initiating**. You need to always exempt the child workflow from the usual initiation process because you are initiating the child workflow in the parent workflow. If you do not change the release status of the child workflow, your workflow will not run as expected. For more information about changing the release status of a workflow, see the help topic Release Status.
- Do not create interdependent workflows that initiate on user events. Doing so can create a logic loop between the workflows and the workflows will not execute as expected.
- Test your workflows to make sure that they execute in the order that you intend them to.

FAQ: SuiteFlow

Review the questions and answers below to learn more information about SuiteFlow. The questions and answers are ordered by topic in the following way:

- Questions About Email
- Questions About Fields
- Questions About Buttons
- Questions About Reports

Questions About Email

The following FAQs describe issues related to SuiteFlow and email.

How do I send an email using SuiteFlow?

Use a delay on a transition to a state that contains a Send Email action or set a delay on the Send Email action itself. See Scheduling an Action, Scheduling a Transition, and Welcome Email Sent to Customers Three Days After First Order Workflow.

Can I include attachments in the Send Email action?

Yes, you can send attachments using the Send Email action. To send multiple attachments in one Send Email action, create a .zip file that includes all the files to send. See Send Email Action.

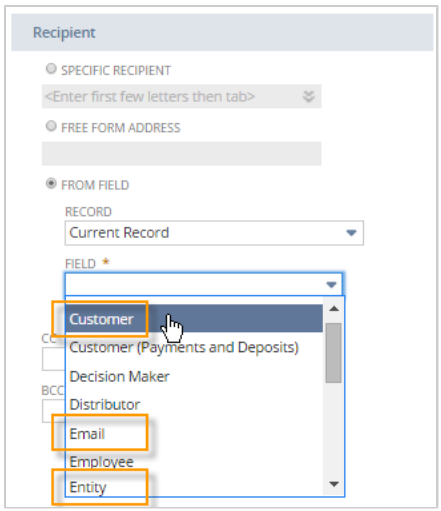
You can also send attachments by creating a custom action using SuiteScript. For more information, see Custom Action. In the custom action, send attachments with email.send(options) or email.sendBulk(options).

In the Send Email action, are Customer, Email, and Entity the same thing?

Yes. A workflow based on a Sales Order, or any other record type that specifies a customer can have the following three values available in the **Field** dropdown list:

- Customer
- Email
- Entity

In this case, for a Send Email action, each of these values reference the same object. If you choose Customer, Email, or Entity, NetSuite uses the email address associated with the customer:



Questions About Fields

The following FAQs describe issues related to fields and SuiteFlow.

How do I set field values using SuiteFlow?

Use the Set Field Value action. See Set Field Value Action.

How do I hide a field using SuiteFlow?

Use the Set Field Display Type action. See Set Field Display Type Action.

How do I make a field required using SuiteFlow?

Use the Set Field Mandatory action. See Set Field Mandatory Action.

Can I set multiselect fields using SuiteFlow?

You cannot set the value of multiselect fields for existing records, either with the Set Field Value action, or with the Send Email, Subscribe To Record, Create Record, Go To Record actions. You can set the value of multiselect fields for new records with Set Field Value.

You can also complete the following other tasks with multiselect fields:

- Change the display type. See Set Field Display Type Action.
- Change the UI label. See Set Field Display Label Action.
- Make the field required. See Set Field Mandatory Action.

How do I set the value of a field based on the value a user set for another field?

Use the Set Field Value action to set the field value after a user edits a different field or set the field value after the user saves the record, depending on the trigger used with the Set Field Value action:

- To dynamically set the field value, use the After Field Edit trigger. See After Field Edit Trigger.
- To set the value after the user saves the record, use Before Record Submit or After Record Submit triggers. See Before Record Submit Trigger or After Record Submit Trigger.

For more information, see Set Field Value Action.

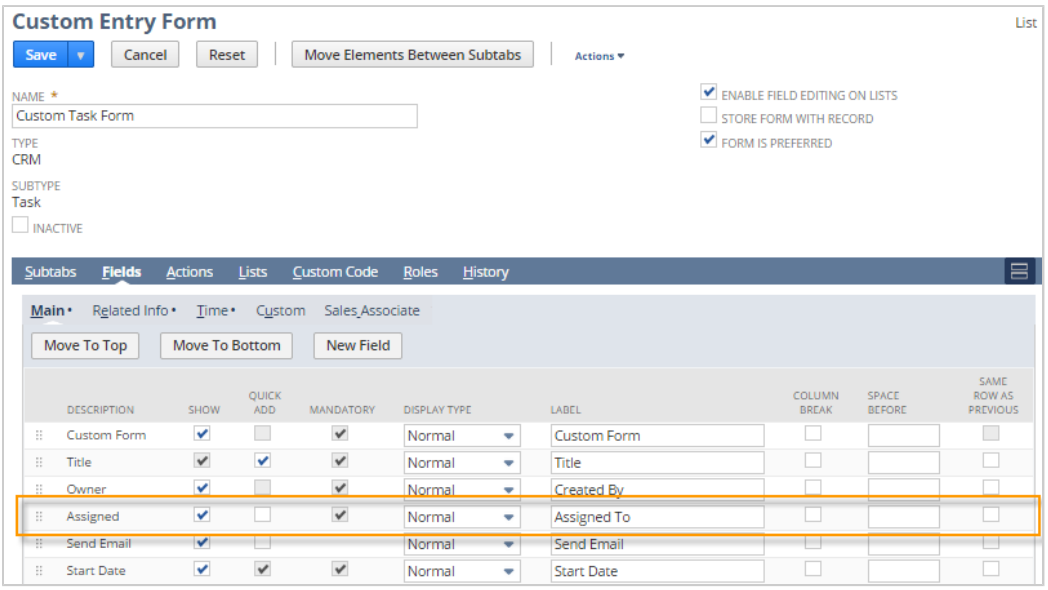
Field names in SuiteFlow dropdown lists are sometimes different than the names of fields that appear in the UI. For example, a field called 'Assigned' in NetSuite may be called 'Assigned To' in SuiteFlow. How do I know if the fields are the same?

View the form customization page for the record that contains the field names. On the form customization page, the **Description** column lists the field names that appear in SuiteFlow, and the **Label** column lists the names that appear in NetSuite.

For example, to access the field names for a Task record:

1. Go to Activities > Scheduling > Tasks > New.
2. In the **Customize** dropdown list, select **Customize Form**.
3. Click the **Fields** subtab.

   On the **Custom Entry Form** page, note the difference in the values between the **Description** column and the **Label** column for the **Assigned** field:



Can I access sublists and sublist fields in a workflow?

No. Sublists and their fields are not accessible from a workflow. To manipulate a sublist in a workflow, create a custom action using a workflow action script. See the help topics N/workflow Module and Custom Action.

Questions About Buttons

The following FAQs describe issues related to buttons and SuiteFlow.

How do I use buttons to help users go to other records and pages in NetSuite?

Use the Add Button action with the Go To Page or Go To Record actions. See Using Buttons for Navigation.

How do I use buttons to transition the record in the workflow from one state to another?

Create the button with the Add Button action and then create a transition that executes on the click of the button. Approval routing workflows use this method to transition between states. See Using Buttons to Execute Transitions.

Why do the buttons I have added to the record disappear when I refresh the browser?

Verify the Add Button actions use the correct trigger for the **Trigger On** property. To properly add buttons to a record form, the trigger must almost always be Before Record Load.

If you use the Entry trigger for the Add Button action, the button is only added the first time a record enters a state. Any actions set to trigger on Entry do not execute again if the record is reloaded during the time that it is in a state. Consequently, the Add Button action executed the first time that the record entered the state, but did not execute again when the record was reloaded. Then the button disappeared.

Use this same guideline with the Remove Button action. Use Before Record Load to make sure that the buttons are always removed, regardless of when and how a user accesses the record during the time that the record is in a specific state.

How do I create a button that, when clicked, navigates users to other records or pages in NetSuite?

See Using Buttons for Navigation. If you want to transition the record in the workflow from one state to another based on a user clicking a button, see Using Buttons to Execute Transitions.

Questions About Reports

Can I generate reports to show which records are in a workflow and how long they have been in the workflow?

Yes. You can search for workflow definitions or workflow instances. Use a workflow search to get information such as which workflows are running, which workflows are inactive, which record types are used in workflows, which employees are using certain workflows, or which workflows came from a bundle installation. See Workflow Searches.

Questions About Errors

The following FAQs describe issues related to errors and SuiteFlow.

If an error occurs (through SuiteFlow or other means) on Before Record Submit, will the After Record Submit actions still be triggered?

No. If a system error occurs between the time a user clicks **Save** on a record and the time that the record is saved to the database, any action set to execute on the After Record Submit trigger does not execute.

Similarly, if a Return User Error action executes on a Before Record Submit trigger, then neither the After Record Submit actions or any unexecuted action on a Before Record Submit trigger do not execute until the user resolves the issues with the record.

What do the "Stack overflow on line <xxx>" or "Too much recursion" errors mean?

These errors occur due to an infinite loop in a workflow. An infinite loop occurs when one operation triggers another operation, and then the second operation triggers the first again.

For example, a workflow uses a Set Field Value action on an After Field Edit trigger and the **Client Field** property in the action definition equal to a specific field on the record. If a user edits the field set for the **Client Field** property, an infinite loop results as the action is continually executed.

An example of an infinite loop in a workflow is a workflow for a purchase order record. The Employee field on a purchase order is dependent on Vendor record, and the list is restricted to employees that belong to the vendor subsidiary. Consequently, the Employee field reloads when a vendor is selected. To avoid this type of infinite loop, set the **Employee != Current User** condition as a workflow condition.

Questions About Actions

The following FAQs describe issues related to actions and SuiteFlow.

Can I copy a workflow?

Yes. Use the Make Copy command, available from the **More** menu on the workflow definition page. See Copying a Workflow.

Can a workflow kick off another workflow?

Yes. Use the Initiate Workflow action to start another workflow from within a parent workflow. When the child workflow completes, the record transitions to the next state in the parent workflow. See Initiate Workflow Action.

How do I create a child workflow?

First, create the parent and child workflows. Then, in the parent workflow, use the Initiate Workflow action to initiate the child workflow. See Creating a Workflow and Initiate Workflow Action.

Can a workflow execute SuiteScript?

Yes. Create a workflow action script and then use a custom action to call the workflow action script. See the help topics SuiteScript 2.x Workflow Action Script Type and Custom Action.

Can workflows be bundled?

Yes. See Bundling a Workflow.

Can the workflow take some action if a record has been in a particular state too long, for example, if the business process is expired?

Yes. Set the Delay field on a transition. See Scheduling a Transition.

If I inactivate an action, will it also apply to records with currently active workflow instances?

Yes, if you inactivate an action, then the action no longer executes, even if an existing workflow instance is currently running.

Can I inactivate a transition?

No, you cannot inactivate a transition.

The Payment Method field on transactions isn't being set. What's wrong?

If a workflow based on the Cash Sale, Customer Deposit, or Customer Payment transaction record types needs to set the **Payment Method** field, make sure the Set Field Value action triggers on the Before User Edit client trigger, instead of the Before Record Load server trigger.

The sourcing of the **Payment Method** field from the customer overrides the default value set by a workflow. Use the Before User Edit trigger to allow all fields to be sourced before the Set Field Value action sets the value of the field.

How do I execute an action on records returned in a saved search?

Create a scheduled workflow to perform a saved search and initiate a workflow instance on each record returned by the saved search. See Scheduling a Workflow.

How do I send statements to customers?

For workflows with a base record type of Customer, use the Send Email action to send statements. For more information, see Attaching Statements to an Email and Attaching Transactions to an Email.

How do I send transactions in a workflow?

For workflows with a base record type of Customer, use the Send Email action to send other transactions. For more information, see Attaching Statements to an Email and Attaching Transactions to an Email.

When I use the Go To Record action in my workflow, why are some of the fields not automatically populated as expected?

If you create a workflow that includes a Go To Record action that sources some field values, the related fields on the new record might not populate automatically. For example, if you have a workflow that runs

on the Task record and automatically populates the Vendor field on Purchase Order records (action A), the system should automatically set the subsidiary based on the subsidiary for the vendor (action B). If you use the Go To Record action in this workflow example, the Go To Record action completes action A but does not complete action B and the subsidiary field on the Purchase Order is left blank. This is a known behavior associated with the Before Record Load trigger of a new record.

To workaround this issue, you need to do the following:

1. Create a custom List/Record field with Vendor set as the target record type. Apply this field to the Purchase order record. To avoid confusion, the custom field can be hidden.
2. Create a new auxiliary workflow that executes on the Purchase order record with the On Create initiation type and the Before Record Load trigger.
3. To populate the Vendor field on Purchase orders, including the subsidiary information, add the Set Field Value action to your workflow. Set the trigger to Before User Edit. If necessary, you can add conditions that determine when the subsidiary information is populated. The Set Field Value action should populate the Vendor field on Purchase Order with the value contained in the field we created in step 1.
4. In your original workflow, edit the Go To Record action so that it populates the field you created in step 1, instead of populating the Vendor field directly.

Now, when the original workflow initiates the Go To Record action, the Vendor field and related fields will be populated automatically.

For more information, see the help topic Set Field Value Action.

In my approval routing workflow, why is my "next approver" not being set?

If employees use the **Employee Center** to access records in a workflow, issues with the **Next Approver** field may occur. Setting a **Next Approver** field requires workflow access to data from a related record, usually the Employee record type.

Most users do not have access to other employee records. Consequently, since the workflow runs with the current user permissions, the workflow cannot access the record types required to properly set the field value. This issue can occur with other record types, for examples, purchase orders.

To resolve this issue, you can enable the **Execute As Admin** property on the workflow definition page. This provides users access to all records in NetSuite and enables the **Next Approver** fields to be set accordingly.

For more information about this property, see Execute As Admin.

How do I create a time-based action? For example, how do I send an email two days after a record has entered a state in the workflow?

Use a delay on a transition to a state that contains a Send Email action or set a delay on the Send Email action itself. See Scheduling an Action, Scheduling a Transition, and Welcome Email Sent to Customers Three Days After First Order Workflow.

Questions About SuiteFlow Terminology

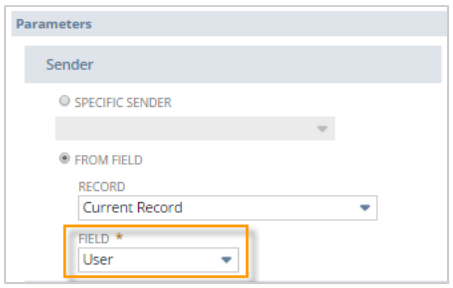
The following FAQs describe issues related to SuiteFlow terminology.

In SuiteFlow, is "user" the same thing as "employee?"

Yes. In SuiteFlow, the term user dynamically defines whoever performed the action that caused a workflow instance to initiate on a record. A user can be an employee, the employee's supervisor, or any other user who performs the action in NetSuite that initiates a workflow instance.

For example, in one state of a workflow, the user might be the employee filling out a travel request. In another state, the user might be the employee's supervisor, who must open the travel request to approve or reject.

The following screenshot shows a Send Email action. In this example, the user in the **From Field** is the user who is currently working with the record, indicated by the **Current Record** value of the **Record** dropdown list.



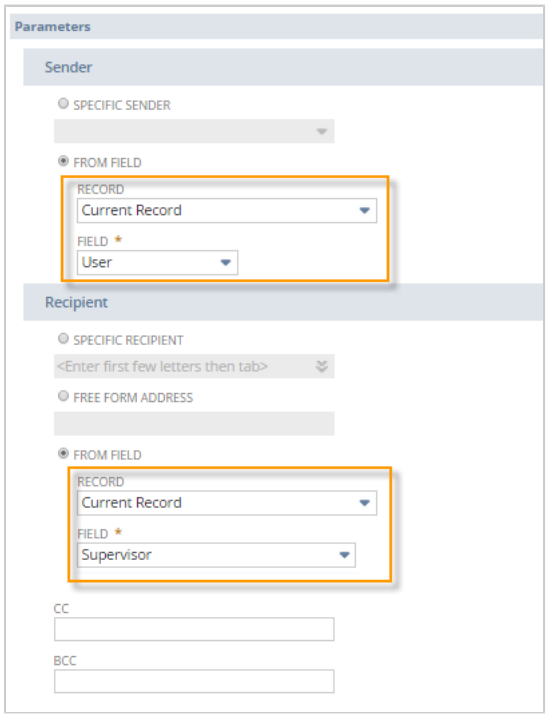
What does "Current Record" mean in SuiteFlow?

The **Current Record** value in the Workflow Manager represents the record in a workflow and the corresponding record values. For example, a user creates a purchase order record and a workflow instance initiates on the purchase order. Use the **Current Record** value to access the values of the purchase order properties in a workflow definition.

The value of the **Current Record** can change dynamically, depending on the user that accesses a record during the time that the record is in a workflow.

Send Email Action Example

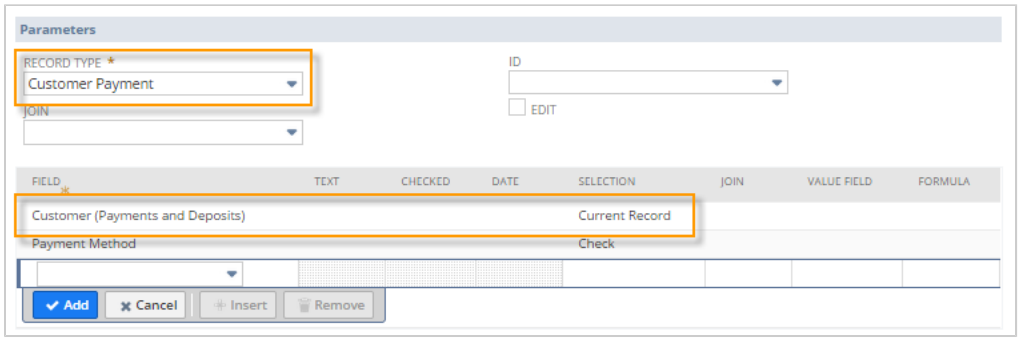
The following screenshot shows a Send Email action for a purchase order. In the **Sender** and **Recipient** sections, the **User** and **Supervisor** values for **Current Record** represent the user currently accessing the purchase order record and the supervisor for that employee, respectively.



Go To Record Action Example

You can also use the **Current Record** values to populate the fields in another record. For example, in a workflow for a Customer record, you use a Go To Record action to create a Customer Payment record. You can use the values on the current Customer record to populate the fields on the new record.

The following screenshot shows the **Parameters** section of a Go To Record action. The **Customer (Payments and Deposits)** field on the Customer Payment record is populated with the value of the Customer entity on the current record.



What is the trigger execution model in SuiteFlow?

For a complete discussion of the SuiteFlow trigger execution model, see the following topics:

- Workflow Triggers and SuiteFlow Trigger Execution Model
- Action Triggers
- Transition Triggers
- Triggers Reference.

Workflow Samples

Lead Nurturing Workflow

The following sample workflow details a lead nurturing campaign created by the fictional company Wolfe Electronics. The Wolfe Electronics marketing department designed this campaign and workflow to market their security software.

The workflow executes the following basic steps after a lead is created in NetSuite, if the lead is part of the software industry:

1. Send an invitation to a webinar. The workflow sends the lead an invitation to a webinar titled "Trends in Internet Security", with a link to an online registration form.

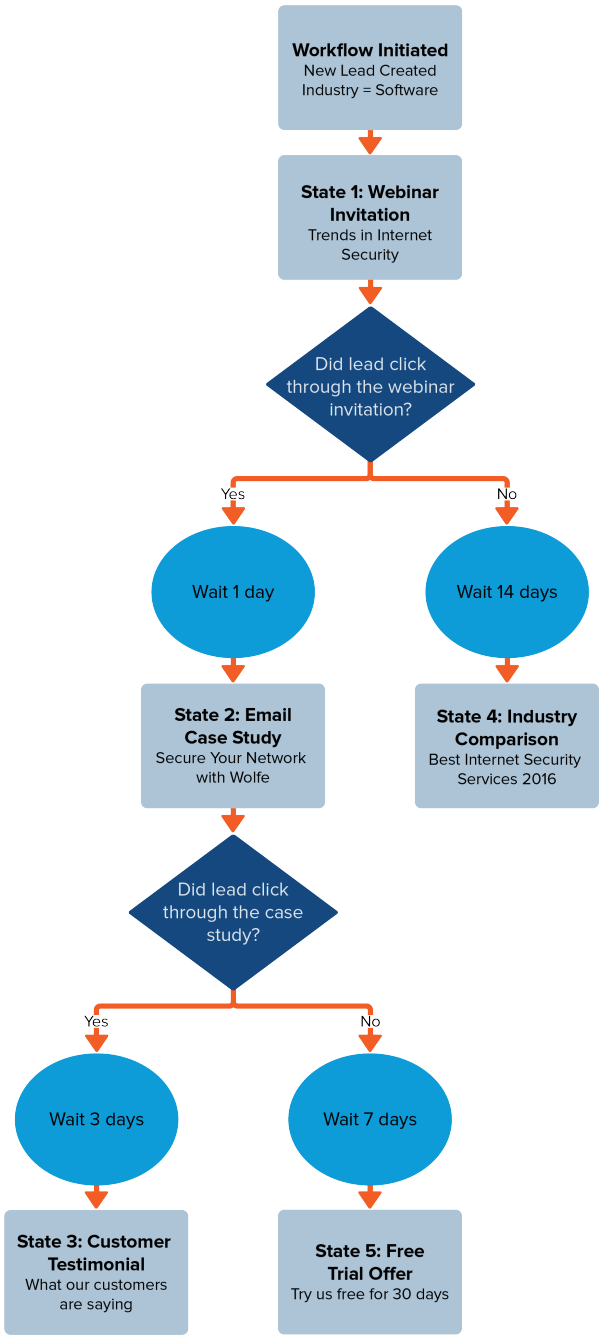
   If the lead does not respond to the webinar invitation in 14 days, the workflow sends an email message titled "Best Internet Security Services of 2014" and the workflow completes.

   If the lead responds to the webinar invitation, the workflow waits one day and proceeds to the next step.
2. Email a case study. The workflow sends an email with link to a case study that describes Wolfe Electronics security offerings.

   If the lead does not click through to the case study, the workflow sends an email 7 days later with a free trial offer and the workflow completes.

   If the lead clicks through to the case study, the workflow waits 3 days and proceeds to the next step.
3. Email a customer testimonial. The workflow sends an email with a customer testimonial and the workflow completes.

The following diagram shows the lead nurturing workflow for Wolfe Electronics:



Steps to Complete the Lead Nurturing Workflow

The following table lists the steps required to complete the lead nurturing workflow:

Step	Description
Designing the Lead Nurturing Workflow	Review the workflow requirements, including the summary of the lead nurturing campaign, workflow definition properties, states, actions, and transitions.
Before You Build the Lead Nurturing Workflow	Create the required marketing templates, lead nurturing campaign record, and Industry custom field to store the recipient of the lead nurturing campaign.

Step	Description
Building the Lead Nurturing Workflow	Create the states and the actions and transitions for each state.
Testing the Lead Nurturing Workflow	Review the steps necessary to test the workflow and then release the workflow.

Note: For more information about the elements of a workflow, see Workflow Elements.

Designing the Lead Nurturing Workflow

Before building a workflow, determine the required workflow states for the campaign. The lead nurturing campaign for Wolfe Electronics requires one state for each lead nurturing campaign event.

Lead Nurturing Campaign Summary

The lead nurturing workflow uses the following workflow definition properties:

Property	Value
Record Type	Customer
Sub Types	Lead
Release Status	Testing*
Initiation	Event Based
Trigger Type	After Record Submit
* Use **Testing** until the workflow is ready for production, then set it to **Released**.	

The following table describes the states and the required actions and transitions:

State	Actions	Transitions
State 1 Webinar Invitation	■ Send Campaign Email: Email a webinar invitation ■ Subscribe To Record: Track response to the email	Transitions to the following states: ■ **State 1a Wait 1 day** if the campaign response is **Clicked Through**. ■ **State 4 Industry Comparison** if the campaign is not responded to after 14 days. ■ **State 6 Exit** if the lead status is **Closed Won**.
State 1a Wait 1 day	There are no actions for this state.	Transitions to the following state: ■ **State 2 Email Case Study**
State 2 Email Case Study	■ Send Campaign Email: Email a case study	Transitions to the following states: ■ **State 2a Wait 3 days** if the campaign response is **Clicked Through**. ■ **State 5 Free Trial Offer** if the campaign is not responded to after 14 days ■ **State 6 Exit** if the lead status is **Closed Won**.
State 2a Wait 3 days	There are no actions for this state.	Transitions to the following state:

State	Actions	Transitions
		▪ **State 3 Customer Testimonia** l
State 3 Customer Testimonial	▪ Send Campaign Email: Email a customer testimonial	Transitions to the following states: ▪ **State 6 Exit** if the lead status is **Closed Won**.
State 4 Industry Comparison	▪ Send Campaign Email: Email an industry comparison	—
State 5 Free Trial Offer	▪ Send Campaign Email: Email a free 30 day trial	—
State 6 Exit	—	—

Next Step: To begin with the lead nurturing workflow example, go to Before You Build the Lead Nurturing Workflow.

Before You Build the Lead Nurturing Workflow

Before you begin building the lead nurturing workflow, complete the following tasks:

- Create the Marketing Templates
- Create the Lead Nurturing Campaign Record and Campaign Events
- Create the Industry Custom Field

Create the Marketing Templates

The lead nurturing workflow uses the Send Campaign Email action to send marketing materials to the customer for the Lead record. Create the templates required by the Send Campaign Email actions.

To create the marketing templates:

1. Go to Lists > Marketing > Marketing Templates > New.
2. Click **Campaign** and then select a template layout.
3. Enter the following properties:

Property	Value
Name	Webinar Invitation
Subject	Trends in Internet Security

4. Upload a template file or enter the template text directly in the Text Editor.
5. On the **Marketing** subtab, select from and reply email addresses, and choose an optional campaign domain.
6. Click **Save**.
7. Repeat the above steps to create the following templates:

Template Name	Subject
Email Case Study	Secure your network with Wolfe Electronics
Customer Testimonial	What our customers are saying
Industry Comparison	Best Internet Security Services 2014
Free Trial Offer	Try us free for 30 days

Create the Lead Nurturing Campaign Record and Campaign Events

Campaign records are used to manage all the information that is important to your marketing efforts. On campaign records, you create events to represent different parts of the same campaign. When you create the Send Campaign Email actions, you select the campaign event associated with the email. See the help topic Managing Campaigns.

To create the lead nurturing campaign record and campaign events:

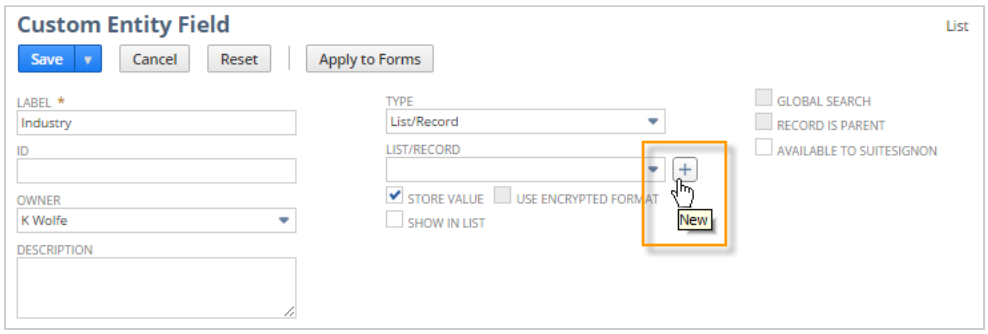
1. Go to Lists > Marketing > Marketing Campaigns > New.
2. In the **Title** field, enter **New Lead Nurturing**.
3. Click the **Lead Nurturing** subtab.
4. In the **Template** column, choose **Webinar Invitation**.
5. If you use the Subscription Categories feature, select a **Subscription** category.
6. Click **Add**.
7. Repeat steps 4 to 6 to add the following templates:
   - Email Case Study
   - Customer Testimonial
   - Industry Comparison
   - Free Trial Offer
8. Click **Save**.

Create the Industry Custom Field

The lead nurturing workflow is designed to send campaign emails to Lead records in the software industry. Create a custom entity field and add it to the Customer record. When a user enters a lead in NetSuite, the user specifies the Industry field and the workflow initiates if the field value is **Software**. For more information about entity fields, see the help topic Creating Custom Entity Fields.

To create the Industry custom field:

1. Go to Customization > Lists, Records, & Fields > Entity Fields > New.
2. In the **Label** field, enter **Industry**.
3. In the **Type** field, select **List/Record**.
4. Next to the **List/Record** field, click **New**.



5. On the **Custom List** page, enter the following properties:

Property	Value
Name	Industry Types
Value	Software

6. Click **Add** and then click **Save**.
7. On the **Custom Entity Field** page, click the **Applies To** subtab, and check the **Customer** box.
8. Click **Save**.

Next Step: To continue with the lead nurturing workflow example, go to Building the Lead Nurturing Workflow.

Building the Lead Nurturing Workflow

To build the lead nurturing workflow, complete the following tasks:

- Create the Workflow Definition
- Create the Workflow Fields

Important: If you have not already done so, complete the prerequisite tasks before you begin this step for the lead nurturing workflow. See Before You Build the Lead Nurturing Workflow.

Create the Workflow Definition

Create the workflow definition and define the basic workflow properties, the workflow initiation, and the workflow initiation condition. The workflow should only initiate after a user saves a new Lead record with an **Industry Type** of **Software**.

To create the workflow definition and define when the workflow initiates:

1. Go to Customization > Workslows > Workflow > New.
2. On the **New Workflow** page, enter the following properties:

Section	Property Name	Value
Basic Information	Name	Lead Nurturing
	Record Type	Customer
	Sub Types	Lead
	Release Status	Testing
Initiation	Event Based	checked
Event Definition	On Create	checked
	On View or Update	unchecked
	Trigger Type	After Record Submit

3. In the **Event Definition** section, select **Visual Builder** and click the **Open** icon to open the Condition Builder.
4. In the **Workflow Condition** window, enter the condition properties in the following columns:

Column Name	Value
Field	Industry
Compare Type	any of
Selection	Software

5. Click **Add**.
6. Click **Save** to save the condition.
7. Click **Save** to save the workflow definition.

Create the Workflow Fields

The workflow executes actions in the workflow states based on the way each recipient responds to the lead nurturing campaign events. The workflow uses workflow fields, based on campaign events responses, to determine the actions and transitions to execute. You must create two workflow fields: **Webinar.Response** and **CaseStudy.Response**.

For more information, see Workflow Custom Fields.

To create the campaign response workflow fields:

1. Make sure the Lead Nurturing workflow is open, created in Create the Workflow Definition.
2. In the context panel, click the **Workflow** tab, click the **Fields** view, and then click **New Workflow Field**
3. On the **Workflow Field** window, enter the following properties:

Property	Description
Label	Webinar.Response
Type	List/Record
List/Record	Campaign Response
Store Value	checked

4. Click **Save**.
5. Repeat the above steps to create another field with the label **CaseStudy.Response**.

Next Step: To continue with the lead nurturing workflow example, go to Creating States for the Lead Nurturing Workflow.

Creating States for the Lead Nurturing Workflow

After you create the workflow definition, create a workflow state for each of the five events in the lead nurturing campaign and an exit state for leads whose records are closed at any point in the workflow. After you create the states, create transitions and then set up each state by creating the actions and setting up the transitions.

To create states for each campaign event:

1. Make sure the Lead Nurturing workflow is open from the step Building the Lead Nurturing Workflow.

2. In the workflow diagrammer, click **New State**.

   A new state appears in the diagrammer.
3. Select the state in the diagrammer and click the **Edit** icon on the **State** tab in the context panel.
4. In the **Name** field, enter **State 1: Webinar Invitation**.
5. Repeat the above steps to create states for the campaign events in the workflow and the exit state:
   - State 1a Wait for 1 day
   - State 2 Email Case Study
   - State 2a Wait for 3 days
   - State 3 Customer Testimonial
   - State 4 Industry Comparison
   - State 5 Free Trial Offer
   - State 6 Exit

   Note: State 6 Exit does not have a section in this sample. It has no actions or transitions because it is the end of the workflow. For more information, see Exit States

Next Step: To continue with the lead nurturing workflow example, go to Creating Transitions for the Lead Nurturing Workflow.

Creating Transitions for the Lead Nurturing Workflow

Transitions define the next step in the workflow for each state. You can create transitions by clicking and dragging in the workflow diagrammer between two states. The diagrammer indicates exit states with the **End** icon.

To create the transitions:

1. Make sure the Lead Nurturing workflow is open from the step Creating States for the Lead Nurturing Workflow.
2. In the diagrammer, hover over the bottom of the **State 1 Webinar Invitation**. The icon becomes a filled half-circle.
3. Drag the icon to the **State 6 Exit**.
4. Release the mouse to create the transition.
5. Repeat the above steps to create the following transitions:

From State	To State
State 1 Webinar Invitation	State 4 Industry Comparison
State 1 Webinar Invitation	State 1a Wait 1 day
State 1a Wait 1 day	State 2 Email Case Study
State 2 Email Case Study	State 6 Exit
State 2 Email Case Study	State 2a Wait 3 days
State 2a Wait 3 days	State 3 Customer Testimonial
State 2 Email Case Study	State 5 Free Trial Offer
State 3 Customer Testimonial	State 6 Exit

> **Note:** The order in which you create the transitions matters. Create the transitions to **State 6 Exit** first, as they are listed on the **Transitions** tab for the state in the order that they are added and execute in that order.

Next Step: To continue with the lead nurturing workflow example, go to State 1 Webinar Invitation.

State 1 Webinar Invitation

After you create the states and transitions, you can set up **State 1 Webinar Invitation**. This state is the point in the workflow where the workflow initiates and the initial email is sent to leads in the software industry. To set up State 1, create a Send Campaign Email action and a Subscribe To Record action and set up the transitions.

To set up State 1 Webinar Invitation:

1. Make sure the Lead Nurturing workflow is open from the step Creating Transitions for the Lead Nurturing Workflow.
2. Select **State 1 Webinar Invitation** in the diagrammer.
3. To create the Send Campaign Email action:
   a. Click **New Action** on the **State** tab in the context panel.
   b. Click **Send Campaign Email** in the **New Action** window.
   c. Enter the following properties:

Property	Value
Transition On	Entry
Recipient	Send To Current Record
Campaign Event	New Lead Nurturing : Trends in Internet Security
Store Result In	Webinar.Response (Workflow)

   d. Click **Save**.

   Since the lead nurturing workflow determines which template to send based on campaign response, subscribe to the campaign response field for the webinar. This way the workflow instance on the lead record triggers each time the campaign response changes.
4. To create the Subscribe To Record action for the campaign response:
   a. Click **New Action** on the **State** tab in the context panel.
   b. Click **Subscribe To Record** in the **New Action** window.
   c. In the **Field** dropdown list, select **Webinar.Response**.
   d. Click **Save**.

   If the lead's record is closed during this workflow, it leaves the workflow and does not receive further email. Each non-exit state in the workflow has a transition to the exit state.
5. To set up the transition to **State 6 Exit**:
   a. In the diagrammer, double-click the transition from **State 1 Webinar Invitation** to **State 6 Exit**.
   b. In the **Workflow Transition** window, select **Visual Builder** and click the **Open** icon to open the Condition Builder.
   c. In the **Workflow Condition** window, enter the following condition properties:

Column Name	Value
Field	Status
Compare Type	any of
Selection	Closed Won

d. Click **Add**.

e. Click **Save** to save the condition.

If the lead clicks on a link in the Webinar Invitation, they move to **State 1a Wait for 1 day** where they will wait for 1 day before moving to **State 2 Email Case Study**.

6. To set up the transition to **State 2 Email Case Study**:

   a. In the diagrammer, double-click the transition from **State 1 Webinar Invitation** to **State 1a Wait 1 day**.

   b. In the **Workflow Transition** window, select **Visual Builder**, click the **Open** icon to open the Condition Builder, and create a condition with the following properties:

Column Name	Value
Record	Webinar.Response
Field	Response
Compare Type	any of
Selection	Clicked Through

   c. Click **Add**.

   d. Click **Save** to save the condition.

7. Click **Save** to save the transition.

   If the lead does not click on a link in the Webinar Invitation, the record moves to State 4 two weeks later.

8. To set up the transition to **State 4 Industry Comparison**:

   a. In the diagrammer, double-click the transition from **State 1 Webinar Invitation** to **State 4 Industry Comparison**.

   b. In the **Workflow Transition** window, set the following properties:

Property	Value
Transition On	Scheduled
Delay	14
Unit	Day

9. Click **Save** to save the changes to the transition.

Next Step: To continue with the lead nurturing workflow example, go to State 1a Wait for 1 day.

State 1a Wait for 1 day

After you set up **State 1 Webinar invitation**, set up **State 1a Wait for 1 day**. This state acts as a placeholder for the workflow during the time that it waits 1 day before it emails the case study. There are no actions associated with the state.

To set up State 1a Wait for 1 day:

1. Make sure the Lead Nurturing Workflow is open from State 1 Webinar Invitation.
2. Select the transition from **State 1 Webinar Invitation** to **State 1a Wait for 1 day** in the diagrammer.
3. In the **Workflow Transition** window, set the following properties:

Property	Value
Transition On	Scheduled
Delay	1
Unit	Day

4. Click **Save**.

Next Step: To continue with the lead nurturing workflow example, go to State 2 Email Case Study.

State 2 Email Case Study

After you set up **State 1a Wait 1 day**, set up **State 2 Email Case Study**. If a lead clicked through the webinar invitation, this state sends an email with a case study to the lead. To set up State 2, create a Send Campaign Email action and a Subscribe To Record action and set up the transitions.

To set up State 2 Email Case Study:

1. Make sure the Lead Nurturing workflow is open from the step State 1a Wait for 1 day.
2. Select **State 2 Email Case Study** in the diagrammer.
3. To create the Send Campaign Email action:
   a. Click **New Action** on the **State** tab in the context panel.
   b. Click **Send Campaign Email** in the **New Action** window.
   c. Enter the following properties:

Property	Value
Trigger On	Entry
Recipient	Send To Current Record
Campaign Event	New Lead Nurturing : Secure your network with Wolfe Electronics
Store Result In	CaseStudy.Response (Workflow)

   d. Click **Save**.

   Since the lead nurturing workflow determines which template to send based on campaign response, subscribe to the campaign response field for the case study.
4. To create the Subscribe To Record action for the campaign response:
   a. Click **New Action** on the **State** tab in the context panel.
   b. Click **Subscribe To Record** in the **New Action** window.
   c. In the **Field** dropdown list, select **CaseStudy.Response (Workflow)**.
   d. Click **Save**.

5. To set up the transition to **State 6 Exit**:
    a. In the diagrammer, double-click the transition from **State 2 Email Case Study** to **State 6 Exit**.
    b. In the **Workflow Transition** window, select **Visual Builder** and click the **Open** icon to open the Condition Builder.
    c. In the **Workflow Condition** window, enter the following condition properties:

Column Name	Value
Field	Status
Compare Type	any of
Selection	Closed Won

    d. Click **Add**.
    e. Click **Save** to save the condition.
6. Click **Save** to save the transition.

   If the lead clicks on a link in the Case Study, they move to **State 2a Wait 3 days** where they will wait 3 days before moving to **Step 3 Customer Testimonial**.
7. To set up the transition to **State 2a Wait 3 days**:
    a. In the diagrammer, double-click the transition from **State 2 Webinar Invitation** to **State 2a Wait for 3 days**.
    b. Select **Visual Builder**, click the **Open** icon to open the Condition Builder, and create a condition with the following properties:

Column Name	Value
Record	CaseStudy.Response (Workflow)
Field	Response Type
Compare Type	any of
Selection	Clicked Through

    c. Click **Add**.
    d. Click **Save** to save the condition.
8. Click **Save** to save the transition.

   If the lead does not click on the link in the case study, the record moves to **State 5 Free Trial Offer** seven days later.
9. To set up the transition to **State 5 Free Trial Offer**:
    a. In the diagrammer, double-click the transition from **State 2 Email Case Study** to **State 5 Free Trial Offer**.
    b. In the **Workflow Transition** window, set the following properties:

Property	Value
Transition On	Scheduled
Delay	7
Unit	Day

    c. Select **Visual Builder** , click the **Open** icon to open the Condition Builder, and create a condition with the following properties:

Column Name	Value
Record	CaseStudy.Response (Workflow)
Field	Response Type
Compare Type	any of
Selection	Received, Sent

d. Click **Add**.

e. Click **Save** to save the condition.

10. Click **Save** to save the changes to the transition.

Next Step: To continue with the lead nurturing workflow example, go to State 2a Wait for 3 days.

State 2a Wait for 3 days

After you set up **State 2 Webinar invitation**, set up **State 2a Wait for 3 days**. This state acts as a placeholder for the workflow during the time that it waits 3 days before it emails the customer testimonial. There are no actions associated with the state.

To set up State 2a Wait for 3 days:

1. Make sure the Lead Nurturing workflow is open from the step State 2a Wait for 3 days.
2. Select the transition from **State 2 Email Case Study** to **Step 2a Wait for 3 days**.
3. In the **Workflow Transition** window, set the following properties:

Property	Value
Trigger On	Schedule
Delay	3
Unit	Day

4. Click **Save**.

Next Step: To continue with the lead nurturing workflow example, go to State 3 Customer Testimonial.

State 3 Customer Testimonial

After you set up **State 2 Email Case Study**, set up **State 3 Customer Testimonial**. If a lead clicked through the case study sent in State 2, this state sends an email with a customer testimonial to the lead. To set up State 3, create a Send Campaign Email action.

To set up State 3 Customer Testimonial:

1. Make sure the Lead Nurturing workflow is open from the step State 2 Email Case Study.
2. Select **State 3 Customer Testimonial** in the diagrammer.
3. To create the Send Campaign Email action:

a. Click **New Action** on the **State** tab in the context panel.
b. Click **Send Campaign Email** in the **New Action** window.
c. Enter the following properties:

Property	Value
Trigger On	Entry
Recipient	Send To Current Record
Campaign Event	New Lead Nurturing : What our customers are saying

d. Click **Save**.

Next Step: To continue with the lead nurturing workflow example, go to State 4 Industry Comparison.

State 4 Industry Comparison

After you set up **State 3 Customer Testimonial**, set up **State 4 Industry Comparison**. If a lead did not click through the webinar invitation, this state sends an industry comparison email. To set up State 4, create a Send Campaign Email action.

To set up State 4 Industry Comparison:

1. Make sure the Lead Nurturing workflow is open from the step State 3 Customer Testimonial.
2. Select **State 4 Industry Comparison** in the diagrammer.
3. To create the Send Campaign Email action:
   a. Click **New Action** on the **State** tab in the context panel.
   b. Click **Send Campaign Email** in the **New Action** window.
   c. Enter the following properties:

Property	Value
Trigger On	Entry
Recipient	Send To Current Record
Campaign Event	New Lead Nurturing : Best Internet Security Services 2014

   d. Click **Save**.

Next Step: To continue with the lead nurturing workflow example, go to State 5 Free Trial Offer.

State 5 Free Trial Offer

After you set up **State 4 Industry Comparison**, set up **State 5 Free Trial Offer**. If a lead did not click through the case study, this state sends a free trial offer email. To set up State 5, create a Send Campaign Email action.

To set up State 5 Free Trial Offer:

1. Make sure the Lead Nurturing workflow is open from the step State 4 Industry Comparison.

2. Select **State 5 Free Trial Offer** in the diagrammer.
3. To create the Send Campaign Email action:
    a. Click **New Action** on the **State** tab in the context panel.
    b. Click **Send Campaign Email** in the **New Action** window.
    c. Enter the following properties:

Property	Value
Trigger On	Entry
Recipient	Send To Current Record
Campaign Event	New Lead Nurturing : Try us free for 30 days

    d. Click **Save**.

Next Step: Now that the workflow setup is complete, test and then release the workflow. To continue with the lead nurturing workflow example, go to Testing the Lead Nurturing Workflow.

Testing the Lead Nurturing Workflow

Before releasing your workflow, test the workflow to make sure it functions as designed.

After setting your workflow release status to Testing, test the following functionality:

- Test workflow initiation. Make sure the workflow initiates with a Lead record is created.
- Test conditions. Set up a testing scenario where you can verify that the conditions in States 1 and 2 can evaluate to both true and false. For more information, see Testing Workflow Conditions.
- Test actions and transitions. Make sure that the actions and transitions in each step appear in the workflow execution log and complete successfully. For more information, see Testing Actions and Transitions.
- Test scheduled transitions. To test the scheduled transitions in Steps 1a and 2a, set the smallest possible delay. For more information, see Testing Scheduled Actions and Transitions.
- Test Send Campaign Email actions. Use the workflow execution log to make sure that the Send Campaign Email actions in Steps 1, 2, 3, 4, and 5 send an email. For more information, see Testing a Send Email Action.

After you have tested the lead nurturing workflow, change the release status to **Released**. See Release Status.

Lead that Did Not Convert to Customer Within Three Days

The following sample shows how to create a workflow that notifies a sales rep if a Lead record did not convert to a Customer record within three days of Lead record creation.

The example uses a workflow field, **Lost Lead Email Sent**, on the lead record form to track if the sales rep was notified. A saved search identifies leads created three days before the current date. The workflow runs the saved search on a schedule and sends the email to the sales rep of the leads that have not been converted to customers in three days that meet the saved search criteria.

To set up the workflow, complete the following steps:

- Create the custom field for the Lead record. See Step 1 Create the Custom Entity Field for the Lead Record.
- Create the saved search. See Step 2 Create the Saved Search.
- Create and schedule the workflow. See Step 3 Create the Workflow and Set the Schedule.

Step 1 Create the Custom Entity Field for the Lead Record

Create a custom entity field for the Lead record form. The field tracks if the sales rep was notified of leads that did not convert. The workflow sets this field to true when the workflow sends the email. This field is set to false for all leads by default.

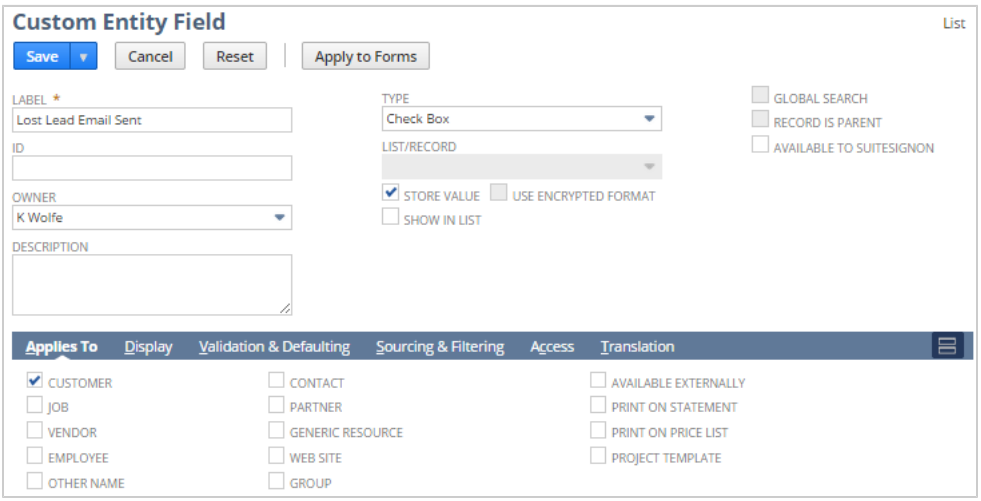
The saved search uses this field as search criteria, so you must create the field before you can create the saved search.

To create the custom entity field for the Lead record form:

1. Go to Customization > Lists, Records, & Fields > Entity Fields > New.
2. On the **New Custom Entity Field** page, enter the following properties:

Property	Value
Label	Lost Lead Email Sent
Type	Check Box

3. On the **Applies To** subtab, enable **Customer**.



4. Click **Save**.

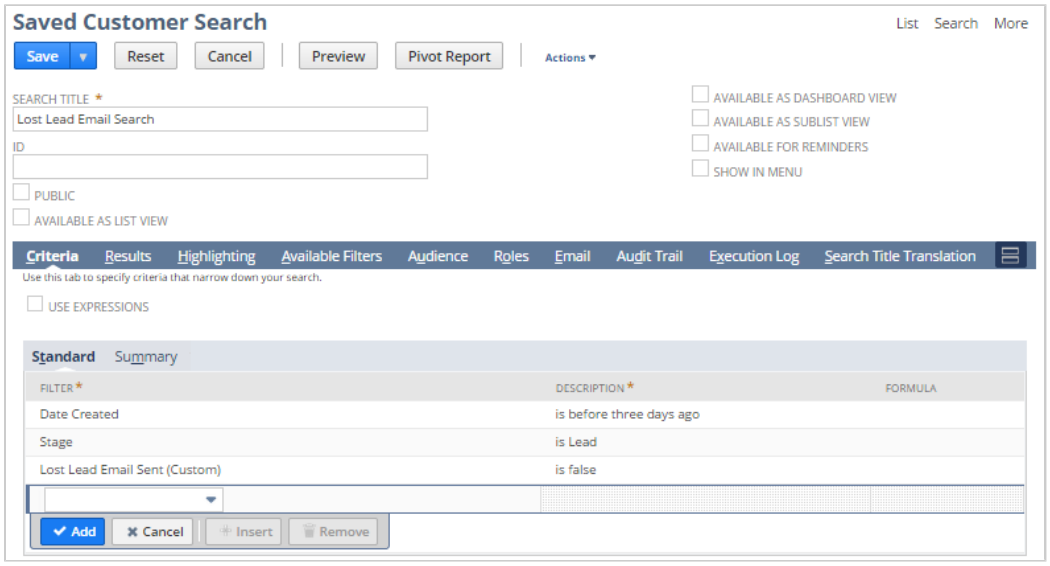
Step 2 Create the Saved Search

Create a saved search to find lead records created more than three days ago with the **Lost Lead Email Sent** field equal to false.

To create the saved search:

1. Go to Lists > Search > Saved Searches > New.
2. On the **New Saved Search** page, click **Customer**.
3. In the **Search Title** field, enter **Lost Lead Email Search**.
4. On the **Criteria** tab, enter the following filter criteria in the **Filter** field:

Filter	Criteria
Date Created	**before** and **3 days ago**
Stage	**any of** and **Lead**
Lost Lead Email Sent (Custom)	No



5. Click **Save**.

Step 3 Create the Workflow and Set the Schedule

Create a workflow for the Lead record and set a schedule for the workflow. Create a single state with a Send Email action and a Set Field Value action for the **Lost Lead Email Sent** field on the Lead record.

To create the workflow and workflow actions:

1. Go to Customization > Workflow > Workflows > New.
2. On the **New Workflow** page, enter the following properties:

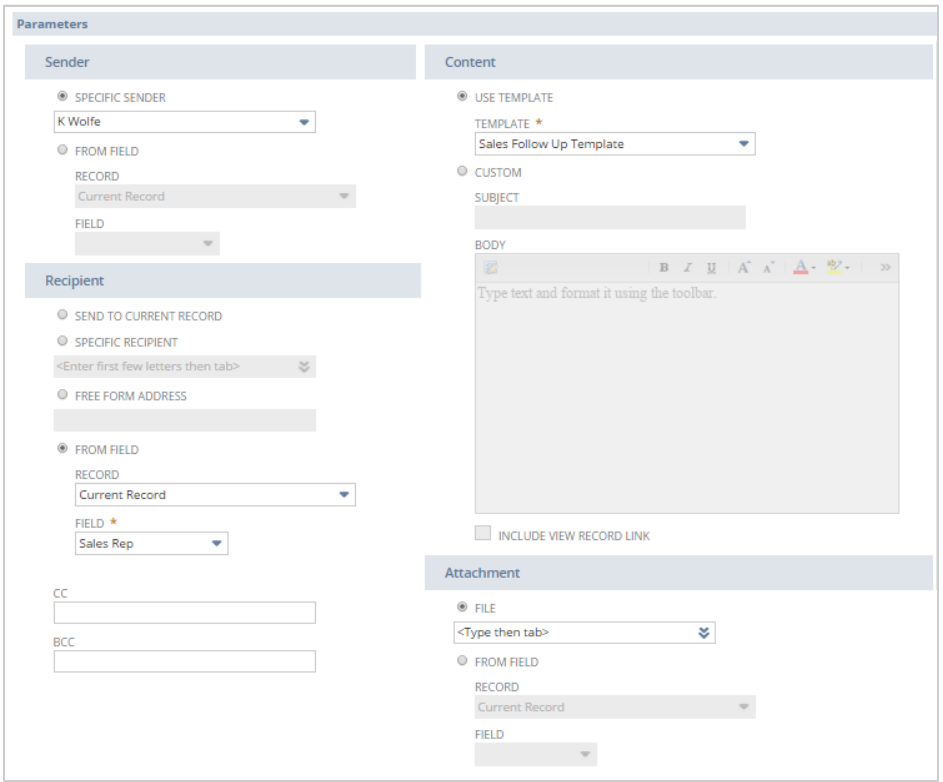
Section	Property Name	Value
Basic Information	Name	Lost Lead Workflow
	Record Type	Customer
	Sub Types	Lead
	Release Status	Released

Section	Property Name	Value
Initiation	Scheduled	checked
Schedule	Saved Search Filter	Lost Lead Email Search
	Repeat	check
	Frequency	Every 30 minutes

3. Click **Save**.
4. The workflow diagrammer, appears with one state.

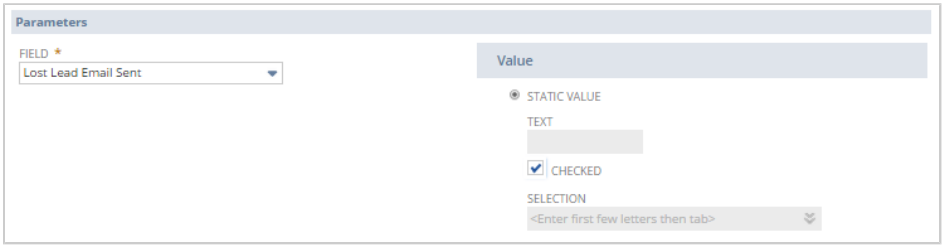
   Select the state in the diagrammer and click the **Edit** icon on the **State** tab in the context panel.
5. In the **Name** field, enter **Send Lost Lead Email**.
6. Click **Save**.
7. To create the Send Email action:
   a. Select the **Send Lost Sale Email** state in the diagrammer.
   b. Click **New Action** on the **State** tab in the context panel.
   c. Click **Send Email** in the **New Action** window.
   d. Enter the following properties:

Section	Property	Value
Sender	Specific Sender	Select a sender.
Recipient	From Field	Record: Current Record Field: Sales Rep
Content	Use Template	Select a template in the **Template** dropdown list.



e. Accept the default values for the other properties and click **Save**.

8. To create a Set Field Value action to set the value of the **Lost Lead Email Sent** field on the lead record to true:
   a. Select the **Send Lost Sale Email** state in the diagrammer.
   b. Click **New Action** on the **State** tab in the context panel.
   c. Click **Set Field Value** in the **New Action** window.
   d. In the **Parameters** section, select **Lost Lead Email Sent** in the **Field** dropdown list.
   e. In the **Value** section, select **Static Value** and enable the **Checked** box.



   f. Accept the default values for the other properties and click **Save**.

Every 30 minutes the saved search runs. If the saved search returns records that match the criteria, NetSuite initiates a workflow instance on each record. The records enter the **Send Lost Sale Email** state and the Send Email action and Set Field Value actions execute. The next time the saved search is run, the saved search will filter out the leads that have the **Lost Lead Email Sent** field set to true.

Note: For more information about how NetSuite runs scheduled workflows, see Scheduled Trigger.

Estimate Approval Routing Workflow

The following sample workflow details approval routing used by the fictional company Wolfe Electronics. This workflow allows supervisors to approve or reject estimates entered by their sales reps.

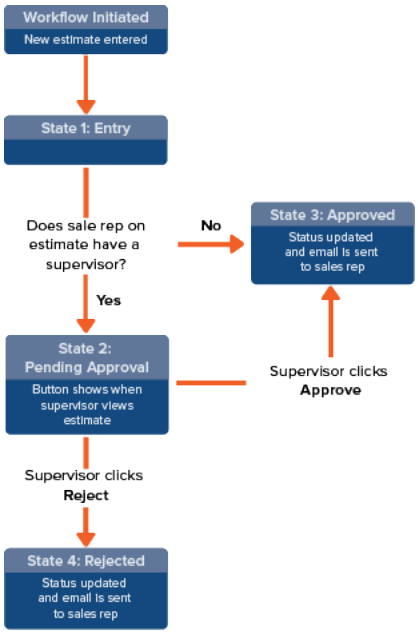
This workflow initiates when a sales rep creates a new estimate record and clicks **Save**. At this point, the estimate is checked to see if the sales rep identified on the estimate has a supervisor on his or her employee record.

The behavior of the workflow depends on the sales rep supervisor:

- Sales rep has a supervisor. The status of the estimate becomes Pending Approval. Buttons to approve or reject the estimate show when the sales rep supervisor views the estimate.

  If the supervisor clicks **Approve**, the status of the estimate becomes Approved and an email is sent to the sales rep. If the supervisor clicks **Reject**, the status of the estimate becomes Rejected and an email is sent to the sales rep.
- Sales rep does not have a supervisor. The status of the estimate becomes Approved and an email is sent to the sales rep.

The following diagram shows the Wolfe Electronics estimate approval workflow:



Steps to Complete the Estimate Approval Workflow

The following table lists the steps required to complete the estimate approval workflow:

Step	Description
Designing the Estimate Approval Routing Workflow	Review the workflow requirements, including the summary of the workflow definition properties, states, actions, and transitions.
Before You Build the Estimate Approval Routing Workflow	Create a custom field to store the approval status for the estimate.
Building the Estimate Approval Routing Workflow	Create the states and the actions and transitions for each state.
Testing the Estimate Approval Routing Workflow	Complete the steps necessary to test the workflow and then release the workflow.

Note: For more information about the elements of a workflow, see Workflow Elements.

Designing the Estimate Approval Routing Workflow

Before building the workflow, determine the workflow states required by estimate approvals. The Wolfe Electronics estimate approval workflow requires one state for each approval status.

Estimate Approval Routing Summary

The lead nurturing workflow uses the following workflow definition properties:

Property	Value
Record Type	Transaction

Property	Value
Sub Types	Estimate
Release Status	Testing*
Initiation	Event Based
Trigger Type	Before Record Submit
* Use **Testing** until the workflow is ready for production, then set it to **Released**.	

The following table describes the states and the required actions and transitions:

State	Actions	Transitions
State 1 Entry	■ Set Field Value: Set the **Approval Status** field to **Pending Approval**	Transitions to the following states: ■ **State 2 Pending Approval** if the estimate shows a default sales rep and that sales rep has a supervisor on their employee record. ■ **State 3 Approved** if the estimate shows a default sales rep and that sales rep has no supervisor on their employee record..
State 2 Pending Approval	■ Add Button: Add an **Approve** button ■ Add Button: Add a **reject** button	Transitions to the following states: ■ **State 3 Approved** if the supervisor clicks the **Approved** button. ■ **State 4 Rejected** if the supervisor clicks the **Reject** button.
State 3 Approved	■ Set Field Value: Set the **Approval Status** field to **Approved** ■ Send Email: Send email to the sales rep	—
State 4 Rejected	■ Set Field Value: Set the **Approval Status** field to **Reject** ■ Send Email: Send email to the sales rep	—

Next Step: To begin with the estimate approval workflow example, go to Before You Build the Estimate Approval Routing Workflow.

Before You Build the Estimate Approval Routing Workflow

Before you begin building the workflow, you must first create the **Approval Status** custom field, required by the Set Field Value actions.

To create the Approval Status custom field:

1. Go to Customization > Lists, Records, & Fields > Transaction Body Fields > New.
2. On the Transaction Body Field page, enter the following properties:

Property	Value
Label	Approval Status

Property	Value
Type	List/Record
Applies To	Sale

3. Click the **New** button next to the **List/Record** field.
4. In the **Name** field of the **Custom Lists** page, enter **Approval Statuses**.
5. On the **Values** tab, enter each of the following values in the **Value** column and click **Add**:
   - Pending Approval
   - Approved
   - Rejected
6. Click **Save** to save the list.
7. Click **Save** to save the custom field.

   The **Approval Status** custom field appears as a dropdown list on the **Custom** subtab of Estimate records.

Next Step: To continue with the estimate approval workflow example, go to Building the Estimate Approval Routing Workflow.

Building the Estimate Approval Routing Workflow

Create the workflow definition and define the basic workflow properties and the workflow initiation properties. The workflow initiates after a user saves a new Estimate record and before NetSuite saves the record data to the database.

To create the workflow definition and define when the workflow initiates:

1. Go to Customization > Workflows > Workflow > New.
2. On the **New Workflow** page, enter the following properties:

Section	Property Name	Value
Basic Information	Name	Estimate Approval Routing
	Record Type	Transaction
	Sub Types	Quote
	Release Status	Testing
	Keep Instance and History	Always
Initiation	Event Based	selected
Event Definition	On Create	checked
	On View or Update	not checked
	Trigger Type	Before Record Submit

3. Click **Save** to save the workflow definition.

Next Step: To continue with the estimate approval workflow example, go to Creating States for the Estimate Approval Routing Workflow.

Creating States for the Estimate Approval Routing Workflow

After you create the workflow definition, create a workflow state for each of the four events in the estimate approval workflow. After you create the states, create transitions and add actions to the states.

To create states for each part of the approval process:

1. Make sure the Estimate Approval Routing workflow is open from the step Building the Estimate Approval Routing Workflow.
2. In the Workflow Diagrammer, double-click **State 1**.
3. In the **Name** field, enter **State 1 Entry**, and then click **Save**.
4. In the Workflow Diagrammer, click **New State**.
5. Double-click the new state.
6. In the **Name** field, enter **State 2 Pending Approval**, and then click **Save**.
7. Create two more states with the following values:
   - **State 3 Approved**
   - **State 4 Rejected**

Next Step: To continue with the estimate approval workflow example, go to Creating Transitions for the Estimate Approval Routing Workflow.

Creating Transitions for the Estimate Approval Routing Workflow

Transitions define the next step in the workflow for each state. In the Workflow Diagrammer, you can create transitions by clicking and dragging between two states. The **End** icon in the Workflow Diagrammer indicates exit states or where the record leaves the workflow.

To create the transitions:

1. Make sure the Estimate Approval Routing workflow is open from the step Creating States for the Estimate Approval Routing Workflow.
2. In the Workflow Diagrammer, hover over the bottom of the **State 1 Entry**. The icon becomes a filled semicircle.
3. Drag the icon onto **State 2 Pending Approval**.
4. Repeat steps 2 and 3 to create the following transitions:

From State	To State
State 1 Entry	State 3 Approved
State 2 Pending Approval	State 3 Approved
State 2 Pending Approval	State 4 Rejected

Note: The order in which you create the transitions matters. Transitions are listed on the Transitions tab in the order that they were added and execute in that order. Create the transition from **State 1 Entry** to **State 2 Pending Approval** first.

Next Step: To continue with the estimate approval workflow example, go to State 1 Entry.

State 1 Entry

After you create the states and transitions, you can set up **State 1 Entry**. This state is the point in the workflow where the workflow sets the initial value of the **Approval Status** field and determines if the estimate requires approval. To set up State 1, create a Set Field Value action and set up the transitions.

To set up State 1 Entry:

1. Make sure the Estimate Approval Routing workflow is open from the step Creating Transitions for the Estimate Approval Routing Workflow.
2. To create the Set Field Value action, do the following:
   a. In the Workflow Diagrammer, click **State 1 Entry**.
   b. In the context panel, on the **State** tab, click **New Action**.
   c. In the **New Action** window, click **Set Field Value**.
   d. Enter the following properties:

Section	Property	Value
Basic Information	Trigger On	Entry
Parameters	Field	Approval Status
Value	Static Value	selected
	Selection	Pending Approval

   e. Click **Save**.
3. To set up the transition to **State 2 Pending Approval** if the sales rep for the estimate has a supervisor, do the following:
   a. In the Workflow Diagrammer, double-click the transition arrow from **State 1 Entry** to **State 2 Pending Approval**.
   b. In the **Workflow Transition** window, in the **Transition On** field, select **Entry**.
   c. In the Condition section, select **Visual Builder**.
   d. Click the **Open** icon to open the Condition Builder, and create a condition with the following properties:

Column Name	Value
Field	Supervisor
Compare Type	not empty

   e. Click **Add**, and then click **Save** to save the condition.
   f. Click **Save** to save the changes to the transition.
4. To set up the transition to **State 3 Approved** if the sales rep for the estimate has no supervisor, do the following:
   a. In the Workflow Diagrammer, double-click the transition arrow from **State 1 Entry** to **State 3 Approved**.
   b. In the **Workflow Transition** window, in the **Transition On** field, select **Entry**.
   c. Click **Save** to save the changes to the transition.

> **Note:** This transition does not need any additional logic, because if the transition to **State 2 Pending Approval** does not execute, then the sales rep has no supervisor and this transition executes instead.

Next Step: To continue with the estimate approval workflow example, go to State 2 Pending Approval.

State 2 Pending Approval

After you set up **State 1 Entry** to determine which estimates require approval routing, set up **State 2 Pending Approval** to enable supervisors to approve or reject estimates. Create the two Add Button actions to add the **Approve** and **Reject** buttons. Then set up the transitions, which depend on which button is clicked when the supervisor views the record.

> **Note:** The **Approve** and **Reject** buttons only appear if the user viewing the record is the supervisor for the sales rep.

To set up State 2 Pending Approval:

1. Make sure the Estimate Approval Routing workflow is open from the step State 1 Entry.
2. To create the Add Button action for the **Approve** button, do the following:
   a. In the Workflow Diagrammer, click **State 2 Pending Approval**.
   b. In the context panel, on the **State** tab, click **New Action**.
   c. In the **New Action** window, click **Add Button**.
   d. In the **Basic Information** section, in the **Trigger On** field, select **Before Record Load**.
   e. In the **Condition** section, select **Visual Builder**.
   f. Click the **Open** icon to open the Condition Builder, and create a condition with the following properties:

Column Name	Value
Record	Sales Rep
Field	Supervisor
Compare Type	any of
Selection	Current User

   g. Click **Add**, and then click **Save** to save the condition.
   h. In the **Parameters** section, in the **Label** field, enter **Approve**.
   i. Click **Save** to save the action.
3. To create the Add Button action for the **Reject** button, do the following:
   a. In the Workflow Diagrammer, click **State 2 Pending Approval**.
   b. In the context panel, on the **State** tab, click **New Action**.
   c. In the **New Action** window, click **Add Button**.
   d. In the **Basic Information** section, in the **Trigger On** field, select **Before Record Load**.
   e. In the **Condition** section, select **Visual Builder**.
   f. Click the **Open** icon to open the Condition Builder, and create a condition with the following properties:

Column Name	Value
Record	Sales Rep
Field	Supervisor
Compare Type	any of
Selection	Current User

g. Click **Add**, and then click **Save** to save the condition.

h. In the **Parameters** section, in the **Label** field, enter **Reject**.

i. Click **Save** to save the action.

4. To set up the transition to **Step 3 Approve** if the supervisor clicks the **Approve** button, do the following:

   a. In the Workflow Diagrammer, double-click the transition arrow from **State 2 Pending Approval** to **State 3 Approve**.

   b. In the **Workflow Transition** window, in the **Execute on Button** field, select **Approve**.

   c. Click **Save** to save the transition.

5. To set up the transition to **Step 4 Reject** if the supervisor clicks the **Reject** button, do the following:

   a. In the Workflow Diagrammer, double-click the transition arrow from **State 2 Pending Approval** to **State 4 Reject**.

   b. In the **Workflow Transition** window, in the **Execute on Button** field, select **Reject**.

   c. Click **Save** to save the transition.

Next Step: To continue with the estimate approval workflow example, go to State 3 Approved.

State 3 Approved

After you set up **State 2 Pending Approval** to enable supervisors to approve or reject estimates, set up **State 3 Approved**. Create a Set Field Value action to set the **Approval Status** to **Approved** and create a Send Email action to notify the sales rep that the estimate was approved.

To set up State 3 Approved:

1. Make sure the Estimate Approval Routing workflow is open from the step State 2 Pending Approval.
2. To create the Set Field Value action, do the following:

   a. In the Workflow Diagrammer, click **State 3 Approved**.

   b. In the context panel, on the **State** tab, click **New Action**.

   c. In the **New Action** window, click **Set Field Value**.

   d. Enter the following properties:

Section	Property	Value
Basic Information	Trigger On	Entry
Parameters	Field	Approval Status
Value	Static Value	selected
—	Selection	Approved

e. Click **Save**.

3. To create the Send Email action, do the following:
   a. In the Workflow Diagrammer, click **State 3 Approved**.
   b. In the context panel, on the **State** tab, click **New Action**.
   c. In the **New Action** window, click **Send Email**.
   d. Enter the following properties:

Section	Property	Value
Sender	From Field	selected
	Record	Sales Rep
	Field	Supervisor
Recipient	From Field	selected
	Record	Current Record
	Field	Sales Rep
Content	Custom	selected
	Subject	Your Estimate Has Been Approved
	Body	Estimate {number} has been approved by your supervisor.
	Include View Record Link	checked

   e. Click **Save**.

Next Step: To continue with the estimate approval workflow example, go to State 4 Rejected.

State 4 Rejected

After you set up **State 3 Approved** to approve estimates and send an email to the sales rep, set up **State 4 Rejected**. Create a Set Field Value action to set the **Approval Status** to **Rejected** and create a Send Email action to notify the sales rep that the estimate was rejected.

To set up State 4 Rejected:

1. Make sure the Estimate Approval Routing workflow is open from the step State 3 Approved.
2. To create the Set Field Value action, do the following:
   a. In the Workflow Diagrammer, click **State 4 Rejected**.
   b. In the context panel, on the **State** tab, click **New Action**.
   c. In the **New Action** window, click **Set Field Value**.
   d. Enter the following properties:

Section	Property	Value
Basic Information	Trigger On	Entry
Parameters	Field	Approval Status
Value	Static Value	selected
	Selection	Rejected

e. Click **Save**.

3. To create the Send Email action, do the following:
   a. In the Workflow Diagrammer, click **State 4 Rejected**.
   b. In the context panel, on the **State** tab, click **New Action**.
   c. In the **New Action** window, click **Send Email**.
   d. Enter the following properties:

Section	Property	Value
Sender	From Field	selected
	Record	Sales Rep
	Field	Supervisor
Recipient	From Field	selected
	Record	Current Record
	Field	Sales Rep
Content	Custom	selected
	Subject	Your Estimate Has Been Rejected
	Body	Estimate {number} has been rejected by your supervisor.
	Include View Record Link	checked

   e. Click **Save**.

Next Step: Now that the workflow setup is complete, test and then release the workflow. To continue with the estimate approval workflow example, go to Testing the Estimate Approval Routing Workflow.

Testing the Estimate Approval Routing Workflow

Before releasing your workflow, it is important to test the workflow to ensure it is functioning as designed.

After setting your workflow release status to Testing, you can verify the following:

- Estimates are transitioning properly.
- Estimate statuses are updated properly.
- Email is being sent in the appropriate state.

After you have tested the estimate approval routing workflow, change the release status to **Released**. See Release Status.

Welcome Email Sent to Customers Three Days After First Order Workflow

The following example shows how to create a scheduled workflow to send a "Welcome" email to a customer three days after the customer places an order.

The example uses a custom field, Email Scheduled, on the customer record form to identify if the email has already been scheduled. A saved search identifies customers who have at least one sales order and

who have not received the welcome email. The workflow runs the saved search on a daily schedule and sends the email to the customers that meet the saved search criteria.

Note: If any current customers have existing sales orders, this workflow also sends the welcome email to those customers.

To set up the workflow, complete the following steps:

1. Create the custom field for the Customer record. For information, see Step 1 Create the Custom Field for the Customer Record.
2. Create the saved search. For information, see Step 2 Create the Saved Search.
3. Create the workflow and set the schedule. For information, see Step 3 Create the Workflow and Set the Schedule.

You can also use these steps to build this process into a larger workflow.

Step 1 Create the Custom Field for the Customer Record

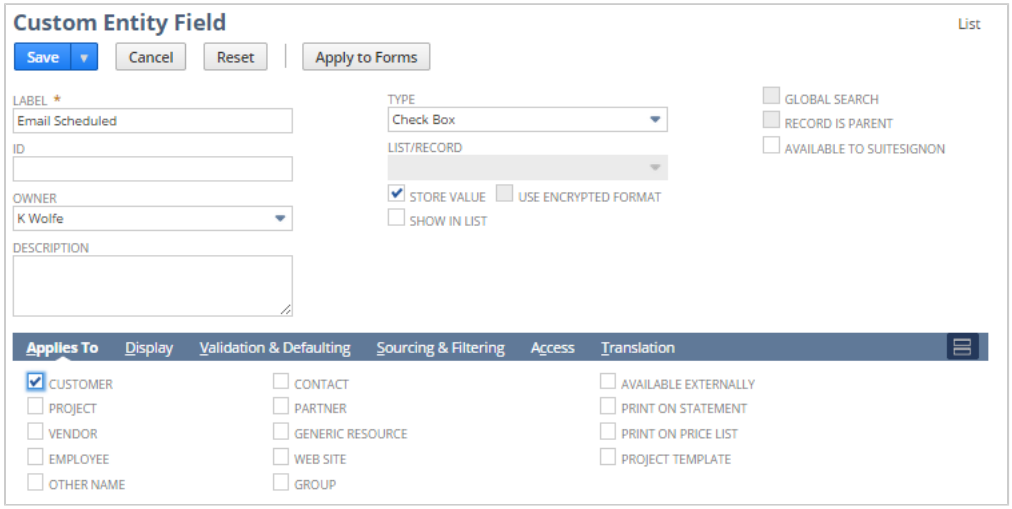
You need to create a custom field for the Customer record form. The field tracks if the welcome email was scheduled. The workflow sets this field to true after NetSuite schedules the email. This field is set to false for all customers by default.

The saved search uses this field as search criteria, so you must create the field before you can create the saved search.

To create the custom entity field for the Customer record form:

1. Go to Customization > Lists, Records, & Fields > Entity Fields > New.
2. On the **New Custom Entity Field** page, enter **Email Scheduled** for **Label** and select **Check Box** for **Type**.
3. On the **Applies To** subtab, enable **Customer**.
4. Accept the default for the other field values.

   The following screenshot shows the **New Custom Entity Field** page:



5. Click **Save**.

This field is initially disabled for all customers, meaning that all customers have not yet received the welcome email.

Step 2 Create the Saved Search

The saved search for this workflow should include following criteria:

- Customers with at least one Sales Order
- Customers with an Email Scheduled value of false

To create the saved search:

1. Go to Lists > Search > Saved Searches > New.
2. On the New Saved Search page, click **Customer**.
3. Enter **Customers with Sales Orders** in the **Search Title** field.
4. To find customers with sales orders:

   On the **Criteria** tab, scroll to the bottom of the **Filter** list, then select **Transaction Fields**. The Saved Customer Search window appears.
5. From the **Transaction Filter** list, select **Type**. The **Type** list appears. Select **any of** from the drop down list, then **Sales Order** from the scrolling list. Click **Set**.
6. On the **Criteria** tab, scroll to the bottom of the **Filter** list, then select **Transaction Fields**. The Saved Customer Search window appears.
7. From the **Transaction Filter** list, select **Main Line**. A list appears. Select **Yes**, then click **Set**.
8. To find customers who have not received the welcome email:

   On the **Criteria** tab, select the **Email Scheduled** field from the **Filter** list. A list appears. Select **No**, then click **Set**.

   The following screenshot shows the **Criteria** tab for the saved search:

9. To find customers with at least one Sales Order, click the **Summary** subtab on the **Criteria** tab and select the following options:
   - From the **Summary Type** list, select **count**.
   - From the **Field** list, select **Transaction Fields**. From the Transaction Field list, select *****, then select **greater than**, and enter a **Value** of **0**. Click **Set**.
10. Click **Save**.

Step 3 Create the Workflow and Set the Schedule

Create a workflow for the Customer record and set a daily schedule for the workflow. You create a single state with a Send Email action and a Set Field Value action for the Email Scheduled field on the Customer record. After you create the workflow, NetSuite runs the saved search a day at midnight and identifies

customers who placed their first Sales Order that day. After three days, the workflow sends the welcome email to those customers.

To create the workflow and workflow actions:

1. Go to Customization > Workflows > Workflow > New.
2. On the **New Workflow** page, enter the following properties:

Section	Property Name	Value
Basic Information	Name	Send Welcome Email
	Record Type	Customer
	Sub Types	Customer
	Release Status	Released
Initiation	Scheduled	checked
Schedule	Saved Search	Customers with Sales Order
	Repeat	check
	Frequency	Daily

   By default, on the Daily schedule, NetSuite runs the saved search every calendar day at midnight.
3. Click **Save**.
4. Select the state in the diagrammer and click the **Edit** icon on the **State** tab in the context panel.
5. In the **Name** field, enter **Welcome Email**.
6. Click **Save**.
7. To create the Send Email action:
   a. Select the **Welcome Email** state in the diagrammer.
   b. Click **New Action** on the **State** tab the context panel.
   c. Click the **Send Email** in the **New Action** window.
   d. In the **Trigger On** field, select **Scheduled**.
   e. Enter the following properties:

Section	Property	Value
Schedule	Delay	3
	Unit	Day
Sender	Specific Sender	Select a sender.
Recipient	From Field	Record: Current Record Field: Customer
Content	Use Template	Select a template from the **Template** dropdown list.

   f. Accept the default values for the other properties and click **Save**.
8. To create a Set Field Value action to set the value of the **Email Scheduled** field on the Customer record to true, complete the following steps:
   a. Select the **Welcome Email** state in the diagrammer.
   b. Click **New Action** on the **State** tab the context panel.
   c. Click **Set Field Value** in the **New Action** window.

d. In the **Parameters** section, select **Email Scheduled** in the **Field** dropdown list.
e. In the **Value** section, select **Static Value** and enable **Checked**.
f. Accept the default values for the other properties and click **Save**.

For testing purposes, you can set the **Delay** on the Send Email action to a smaller time period and use a test email address to verify the email content before you send the email to customers. For more information, see Testing Scheduled Actions and Transitions and Testing a Send Email Action.

Storing a Return Value from a Custom Action Script in a Workflow Field

This sample shows how to store a return value from a custom action script into a workflow field. This example can be useful in the following cases:

1. You want to get a value from the Item sublist and use this value as a condition in the workflow. You use obj.getSublistValue in the script and return this in the workflow.
2. You want to check if a certain item is existing in the Item sublist. The script returns "0" if item is not existing and "1" if it does.
3. You want to make sure that all items in the Item sublist have a quantity equal to or greater than 1 (similar case as #2).

To store a return value from a custom action script in a custom field:

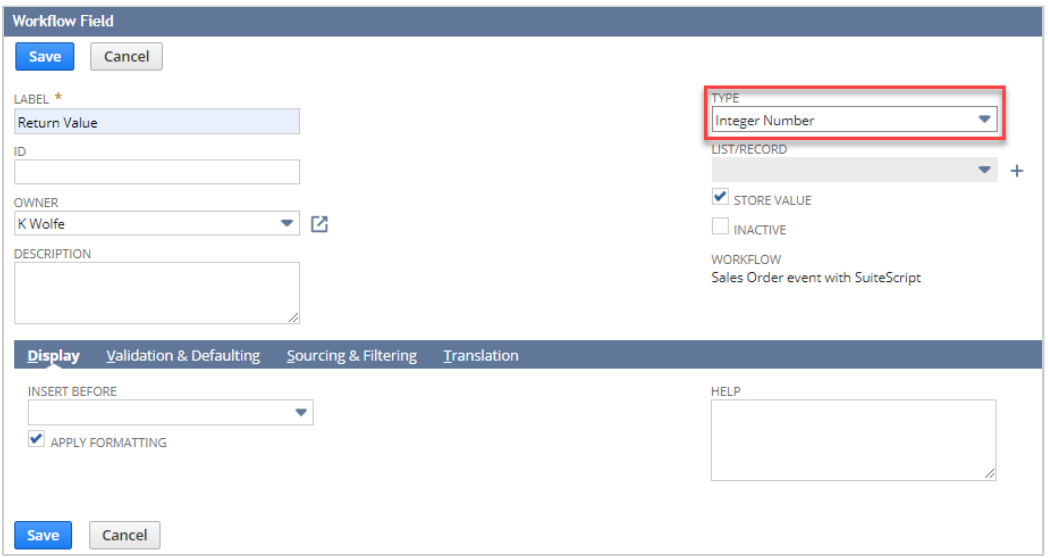
1. Create a new Workflow Action script.

   This script is an example.

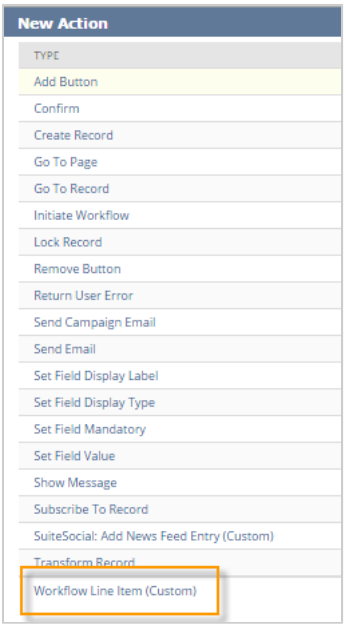
```
/**
 * @NApiVersion 2.x
 * @NScriptType WorkflowActionScript
 */
define([],
    function() {
        function onAction(scriptContext) {
            log.debug({
                title: 'Start Script'
            });
            var newRecord = scriptContext.newRecord;
            var itemCount = newRecord.getLineCount({
                                sublistId: 'item'
                                });
            log.debug({
                title: 'Item Count',
                details: itemCount
            });
            for(var i = 0; i < itemCount; i++)
            {
                var quantity = newRecord.getSublistValue({
                                    sublistId: 'item',
                                    fieldId: 'quantity',
                                    line: i
                                });
                log.debug({
                    title: 'Quantity of Item ' + i,
                    details: quantity
                });
                if(quantity === 0)
                {
                    return 0;
                }
```

```
            }
            log.debug({
                title: 'End Script'
            });
            return 1;
        }
        return {
            onAction: onAction
        }
    });
```

2. Make sure that the script returns a value. You can specify this on the **Parameters** tab of the Script record page.
3. In SuiteFlow, create a workflow field. The field should be the of the same type as the return parameter of the Workflow Action script.



4. Within a state, add the custom action (this is the Workflow Action script).



The return value from the Workflow Action script can be stored in the **Store Result In** field.


Parameters
STORE RESULT IN
Return Value (Workflow)
FIELD
VALUE
JOIN
VALUE FIELD
FORMULA
No records to show.